Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Escuela Técnica Superior de Ingenieros Industriales y de Telecomunicación
Telekomunikazio eta Industria Ingeniarien Goi Mailako Eskola Teknikoa

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

Titulación:

INGENIERO INDUSTRIAL

Título del proyecto:

DISEÑO DE UN SISTEMA DE GESTIÓN REMOTA DE CLIMATIZACIÓN BASADO EN ARDUINO

Judith Flores Fernández

Ángel María Andueza Unanua

Pamplona, 24 de Abril de 2015

RESUMEN

El objetivo del proyecto consiste en el desarrollo de un sistema para el control de la temperatura basado en la plataforma Arduino. Se implementará un sistema que no sólo permita la consulta y el control de la temperatura del hogar desde el interior de este, sino también desde el exterior, a través de cualquier dispositivo con acceso a internet.

Se pretende que el sistema final sea útil para varios tipos de calderas cuyos sistemas de actuación son distintos; por lo que el prototipo a crear será modular, de modo que se pueda conectar el módulo adecuado en cada vivienda. Así pues, se crearán varios módulos de actuación, al mismo tiempo que varios módulos de sensado de temperatura basados en distintos sensores.

Finalmente, se creará en el entorno de desarrollo Arduino un servidor web que permita controlar la calefacción de manera remota modificando la temperatura de consigna del termostato; así como consultar la temperatura ambiente de la vivienda y el estado de la calefacción.

ABSTRACT

The main objective of this project is to develop a temperature control system based on Arduino. The implemented system will allow not only to check and control home temperature from the inside of the house, but also from the outside of it, using any device with access to the internet.

The final system is intended to be useful for various types of heaters which are activated in different ways. Therefore, the prototype will be modular, so it will be possible to connect the appropriate module depending on the heating system of each house. Hence, different control modules will be developed, as well as various temperature sensing modules based on diverse sensors.

Finally, a web server will be developed on Arduino environment, so it is possible to control the heating system by modifying the desired thermostat temperature; as well as to check the house temperature and the heater state.

ÍNDICE

ÍNDICE DE FIGURAS

1. INTRODUCCIÓN

1.1 OBJETIVO

El objeto de este proyecto es el diseño de un sistema de control remoto de climatización basado en el hardware Arduino y en programación html. El interés personal por la domótica, así como por todas sus posibles aplicaciones, son los principales motivos de esta elección.

La domótica se entiende como el "conjunto de técnicas orientadas a automatizar una vivienda, que integran la tecnología en los sistemas de seguridad, gestión energética, bienestar y/o comunicaciones". Como se puede deducir de esta definición, la domótica comprende la gestión y control de una gran variedad de sistemas y dispositivos dentro de una vivienda o un edificio. Por lo tanto, el desarrollo de un sistema de domótica general que actúe sobre distintas unidades de una casa, tales como iluminación (tanto apagado y encendido como control de nivel de luz), riego, climatización, calderas, toldos, persianas, alarmas, etc., supone un proyecto muy amplio y ambicioso, demasiado para el periodo de tiempo del que se dispone para su realización. Se obtendría un sistema con aplicaciones muy extensas, pero con poca profundización y especialización en cada una de las partes.

En consecuencia, y a fin de realizar un trabajo más exhaustivo y en más profundidad, se decide acotar el rango de actividades a controlar y basar el proyecto únicamente en una de ellas: el control de climatización.

Uno de los motivos que han dado lugar a esta elección es el hecho de que, a diferencia de otras aplicaciones de la domótica, la climatización es un sistema sobre el cual resulta muy útil e interesante tener control desde fuera de la vivienda, para conseguir que ésta se encuentre a la temperatura deseada en el momento en que está previsto regresar al hogar.

Así, se pretende crear un prototipo de termostato que, mediante el uso de la electrónica, programación en el entorno de desarrollo Arduino y programación html permita al usuario controlar la climatización desde cualquier lugar, tanto dentro como fuera de la vivienda, a través de cualquier dispositivo móvil con acceso a internet, como puede ser un Smartphone, un PC o una Tablet.

Para comenzar a desarrollar este proyecto es necesario, primeramente, concretar el tipo de sistema de climatización que se ha de controlar. A la hora de definir este proyecto, se decide realizar un control de calefacción, ya que es el sistema de climatización más empleado, por encima de otros como aires acondicionados o ventiladores.

Sin embargo, dentro de los sistemas de calefacción, existen diversos tipos según cual sea el tipo de suministro de energía empleado. Actualmente, entre los sistemas empleados destacan los siguientes:

- Calefacción eléctrica: una corriente eléctrica recorre una resistencia, la cual disipa energía en forma de calor, convirtiendo la energía eléctrica en calor. Este principio puede ser empleado directamente en radiadores eléctricos, los cuales irradian el calor generado al ambiente, o bien en calderas eléctricas, las cuales emplean el calor generado para calentar agua. Dentro de la calefacción eléctrica se encuentran también los acumuladores de calor. Estos aparatos constan de ladrillos de arcilla u otro material cerámico, los cuales se calientan durante la noche a través de resistencias u otros elementos de calentamiento eléctrico empotrados en ellos.

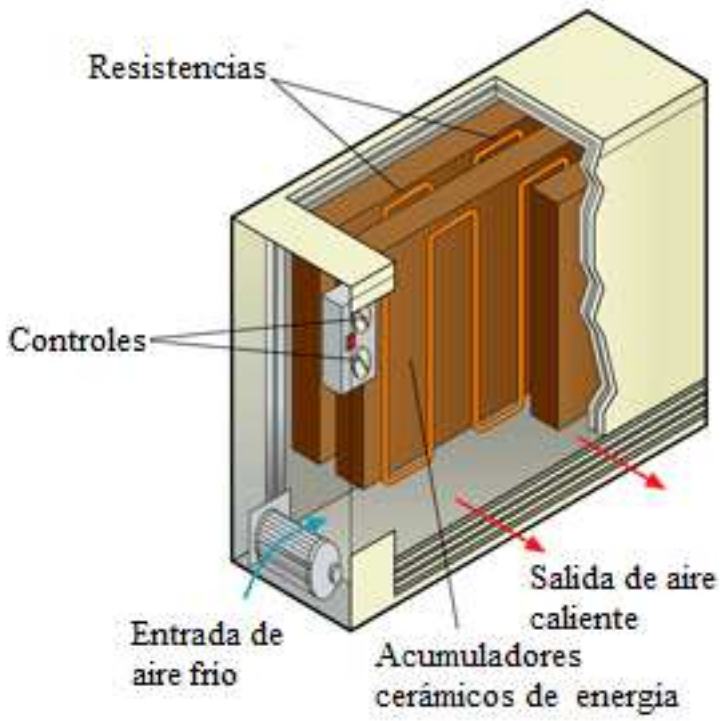


FIG. 1: INTERIOR DE UN ACUMULADOR DE CALOR

Durante el día, el calor almacenado se desprende lentamente produciendo un intercambio de calor con el ambiente. Además, habitualmente, los acumuladores disponen de ventiladores para acelerar el intercambio de calor.

- Calefacción por gas: la parte fundamental de estos sistemas es la caldera de gas natural. Esta caldera calienta agua y, a través de un sistema de tubos, distribuye ese agua caliente por los radiadores de la vivienda, de modo que la diferencia de temperaturas entre el agua del interior del radiador y el ambiente del exterior hace que se produzca una transmisión de calor, aumentando así la temperatura de la vivienda. Tras recorrer el radiador, el agua, ya fría, vuelve a la caldera para volver a ser calentada.

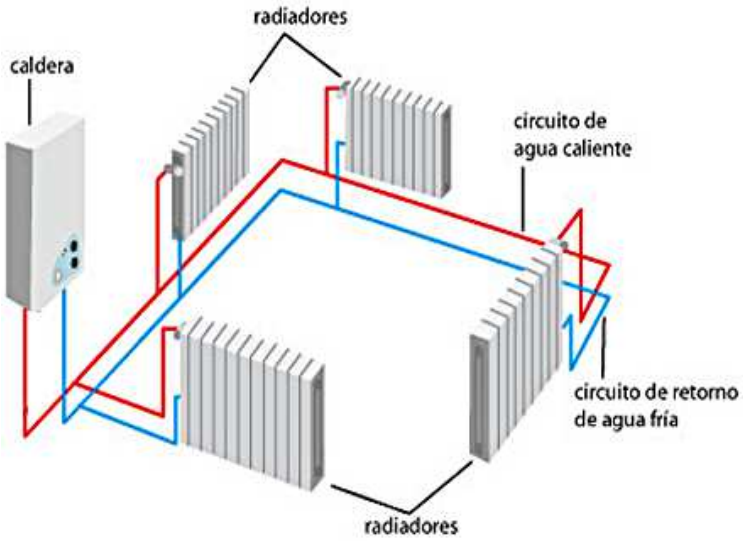


FIG. 2: ESQUEMA GENERAL DE UN SISTEMA DE CALEFACCIÓN POR GAS

En la actualidad, aunque siguen existiendo viviendas con instalaciones de calefacción eléctrica, éstas son una minoría, y la calefacción por gas se ha impuesto a todos los demás sistemas, debido principalmente a dos razones: la expansión de las redes de gas natural, haciéndolo más accesible; y la elevada eficiencia de este tipo de calefacción.

Como consecuencia de todo lo anteriormente explicado, se opta por centrar este proyecto en los sistemas de calefacción por gas; ya que son los más extendidos y, por tanto, el sistema que se desarrolle será útil en un mayor número de viviendas.

Una vez elegido el tipo de calefacción, es necesario escoger los dispositivos de transmisión de calor a emplear. Actualmente, las calderas de gas funcionan bien con radiadores o con suelo radiante. Los radiadores son los más utilizados y se colocan directamente en la pared sin necesidad de obras. El suelo radiante, sin embargo, tiene todavía una baja tasa de instalación en viviendas ya que se trata de un sistema relativamente moderno que, además, requiere una mayor y más compleja mano de obra que los radiadores convencionales en cuanto a instalación se refiere.

Teniendo esto en cuenta, se decide basar el proyecto en radiadores convencionales ya que, al igual que ocurre con las calderas de gas, esto hará posible implantar el sistema desarrollado en un mayor número de viviendas.

Así pues, el proyecto queda definido y acotado a sistemas de calefacción que empleen calderas de gas y radiadores convencionales.

1.2 ESTADO DEL ARTE

El termostato es el componente principal de los sistemas de control de climatización, ya que se encarga de abrir o cerrar un circuito eléctrico en función de la temperatura, activando y desactivando las calderas. El sistema a desarrollar en este proyecto pretende ser una continuación de los actuales termostatos empleados para el control de temperatura.

Actualmente, existen diversos tipos de termostatos, los cuales pueden ser clasificados en varios grupos según su principio de funcionamiento.

1.2.1 TERMOSTATOS MECÁNICOS

Este tipo de termostatos sensan y controlan utilizando únicamente medios puramente mecánicos.

Dentro de esta clase cabe destacar, por su utilización en sistemas de climatización, los termostatos bimetálicos, los cuales incluyen dos metales unidos formando lo que se llama un bimetal. Uno de los extremos del bimetal se fija a una fuente de tensión y al otro se le añade un contacto eléctrico. Al tratarse de metales distintos, sus coeficientes de expansión térmica también son diferentes y, en consecuencia, cuando la temperatura del ambiente sube, uno se expande más que otro, haciendo que el bimetal se doble tal y como se muestra en la siguiente imagen.

FIG. 3: INTERIOR DE UN TERMOSTATO MECÁNICO

De este modo, los metales se alejan del contacto del termostato dejando el circuito abierto y apagando por tanto el aparato que esté conectado a éste (generalmente una caldera). Conforme la temperatura disminuye una vez se ha apagado la calefacción, los metales se contraen de nuevo hasta volver a tocar el contacto del termostato, volviendo a encender la caldera al cerrar el circuito.

El bimetal no se expande ni contrae repentinamente, sino que éste es un proceso que requiere cierto tiempo, por lo que la calefacción no está continuamente encendiéndose y apagándose; lo cual provocaría deterioro en el sistema, además de ser poco eficiente.

En este tipo de termostatos el control de la temperatura de consigna se hace mediante un interruptor giratorio. Al girar el selector de temperatura, éste mueve el contacto de modo que se acerque más a la lámina de bimetal si la temperatura fijada es

mayor y viceversa. Por tanto, cuanto mayor sea la temperatura de consigna, más curvado estará el bimetal en el momento en que haga contacto.

Los termostatos mecánicos sólo permiten un control ON/OFF de las calderas, ya que o bien el bimetal y el resto del circuito están en contacto, cerrando el circuito, y haciendo que la caldera esté encendida, o bien están separados, dejando el circuito abierto y apagando por tanto la caldera.

1.2.2 TERMOSTATOS DIGITALES

Los termostatos más modernos ya no se basan en principios mecánicos, sino que utilizan termistores y otros tipos de elementos semiconductores para determinar la temperatura ambiente. La temperatura de consigna ya no es fijada mediante un interruptor giratorio sino a través de pulsadores (uno para subir la temperatura y otro para bajarla). En consecuencia, estos termostatos necesitan de la instalación de pilas, ya que los elementos electrónicos que emplean deben ser alimentados para que puedan funcionar.

Además, los termostatos digitales tienen una pantalla LCD en la que, generalmente, se muestra la temperatura actual de la habitación donde éste se encuentra situado; así como la temperatura de consigna fijada por el usuario. Muchos también disponen de un reloj interno que permite programar horarios de encendido de la calefacción, para proporcionar tanto mayor confort como mayor ahorro de energía (alrededor de 30%).

FIG. 4: TERMOSTATO DIGITAL

Al contrario que los termostatos mecánicos, los termostatos digitales no solo permiten un control ON/OFF, el cual se realiza a través de relés o contactos libres de potencial, sino que también permiten controles regulables. Éste último tipo de control se puede llevar a cabo bien mediante el uso de electroválvulas controladas por un servomotor que ajuste la temperatura del agua de calefacción, o mediante regulación analógica configurada directamente en la programación del microcontrolador del termostato. Estos tipos de controles o sistemas de actuación serán expuestos con más detalle en apartados posteriores.

1.2.3 CONTROL DE ENCENDIDO DE CALDERAS MEDIANTE SMS

Los controladores de encendido de calderas mediante SMS o controladores GSM han surgido como evolución de los termostatos digitales. Tienen las mismas funciones que estos últimos (control de temperatura mediante temperatura de consigna y programación horaria) pero incorporan una característica más: pueden ser controlados mediante SMS. Tanto la temperatura de consigna como los horarios de encendido y apagado pueden ser fijados a través de un mensaje de texto utilizando un código muy sencillo.

Estos controladores incorporan una tarjeta SIM idéntica a las de los teléfonos móviles de modo que, cuando un SMS es enviado a ese número de teléfono, el aparato lo recibe y es capaz de descodificarlo y convertirlo en una acción de control (encender caldera, apagar caldera, encender o apagar caldera un día a una hora determinada, etc.). El aparato posee un interruptor interno que se cierra en el momento en que le llega la orden de encender, de modo que cierra el circuito haciendo que llegue corriente a la caldera y que, por tanto, esta empiece a funcionar.

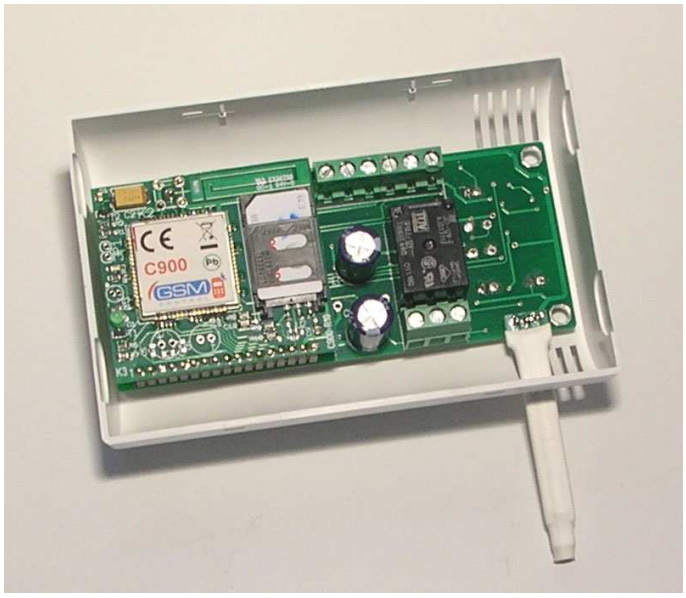


FIG. 5: TERMOSTATO GSM

El desarrollo de este aparato ha supuesto un gran avance en el control de climatizaciones ya que, tal y como se tiene por objetivo en este proyecto, permite la manipulación del termostato de manera remota. Sin embargo, como se ha explicado anteriormente, los controladores GSM son capaces únicamente de abrir o cerrar un contacto de modo que llegue corriente a la caldera en el momento en que se desea que ésta esté encendida. Por tanto, solamente es útil para calderas activadas mediante con controles ON/OFF, y no para sistemas regulables o modulantes.

1.2.4 CONTROL 3G WIFI DE CALDERAS

Se trata del termostato más moderno y completo actualmente disponible. El termostato consta de conexión a internet de modo que, programando el microcontrolador, datos como la temperatura actual, la temperatura de consigna, el

estado actual de la caldera (ON/OFF), etc., puedan ser enviados a una aplicación disponible para Android y PC. Así pues, desde dispositivos móviles como Smartphones o computadoras, se pueden tanto leer como modificar estos datos, consiguiendo así un control remoto de la climatización de la vivienda.

FIG. 6: TERMOSTATO 3G WIFI

Se trata pues de un sistema muy similar al que se pretende desarrollar en este proyecto. Sin embargo, de momento sólo está disponible para calderas de la marca Ducasa y únicamente permite el control de calderas no modulantes, activadas mediante relé u optoacopladores. Además, su precio actual de venta es muy elevado (rondando los 200€).

En consecuencia, se pretende que el producto final de este proyecto sea una evolución de este tipo de termostatos; pero que no sólo sirva para un tipo de calderas de una marca concreta, sino que sea apto para todos los tipos y marcas, mediante un pequeño y sencillo ajuste de software en el momento de la instalación. Además, otro de los principales objetivos es reducir el coste del producto para que sea más asequible, ya que los termostatos 3G Wi-Fi actuales cuestan prácticamente una tercera parte del precio de una caldera básica.

1.2.4 TERMOSTATOS BASADOS EN ARDUINO

Arduino es una plataforma de software libre, por lo que los diseños y prototipos basados en él no se pueden comercializar. Por esta razón, no existen en el mercado termostatos en base a placas Arduino. Sin embargo, sí que se pueden encontrar en internet sistemas diseñados por usuarios de Arduino tanto aficionados como profesionales para su uso personal en viviendas. Algunos de estos diseños incluyen control remoto y muchos de ellos no.

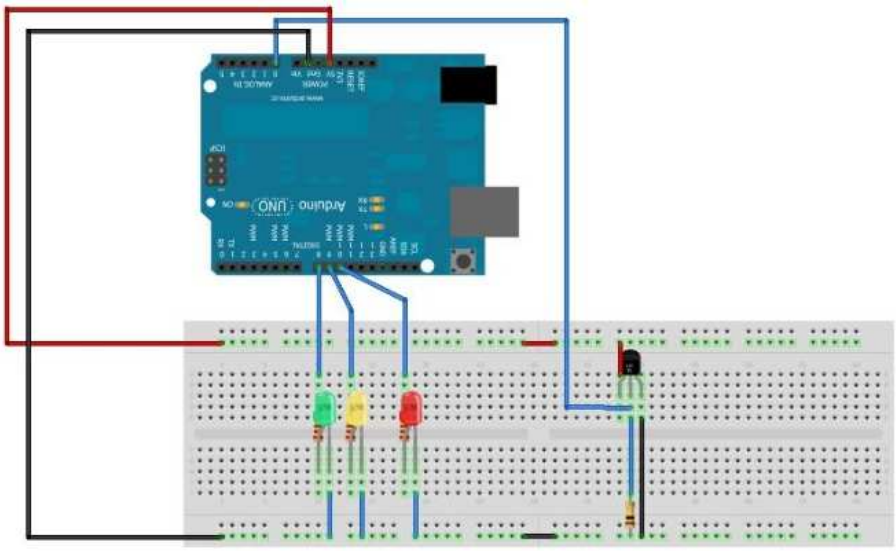


FIG. 7: EJEMPLO DE TERMOSTATO BÁSICO CON ARDUINO

Este es pues el punto de partida del presente proyecto. Se creará un prototipo para su uso en viviendas particulares pero, en lugar de estar diseñado para una vivienda en particular con una caldera determinada, como los creados hasta el momento, se hará de manera que el termostato final pueda ser empleado en distintas viviendas con diferentes sistemas de calefacción.

2. PROPUESTA DE DISEÑO DEL PROTOTIPO

Como ya se ha comentado en el resumen, el principal objetivo de la creación de este prototipo es que sea útil para varios tipos de calderas; por lo que lo esencial será crear diferentes módulos que puedan ser conectados al dispositivo final según sean necesarios o no.

Al igual que cualquier termostato, el prototipo constará de diferentes partes o etapas:

- Sensado de temperatura
- Actuación (control de la caldera)
- Control remoto (ajuste de temperatura de consigna de manera remota)

Así pues, el sistema de control quedará como sigue:

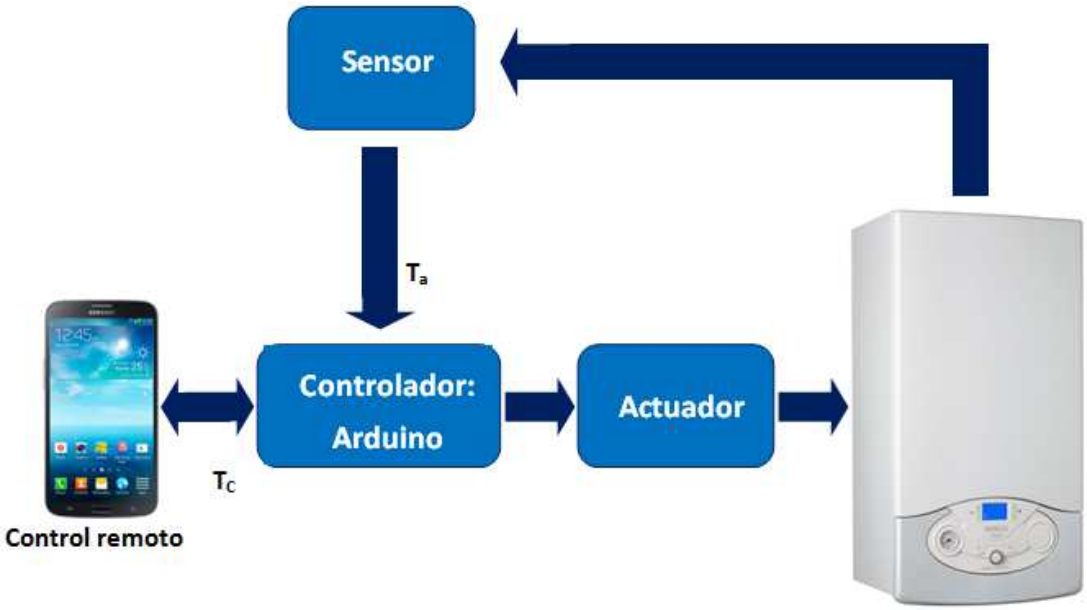


FIG. 8: LAZO DE CONTROL

Además, cada una de estas etapas podrá ser llevada a cabo por distintos módulos; los cuales se especifican en la siguiente imagen:

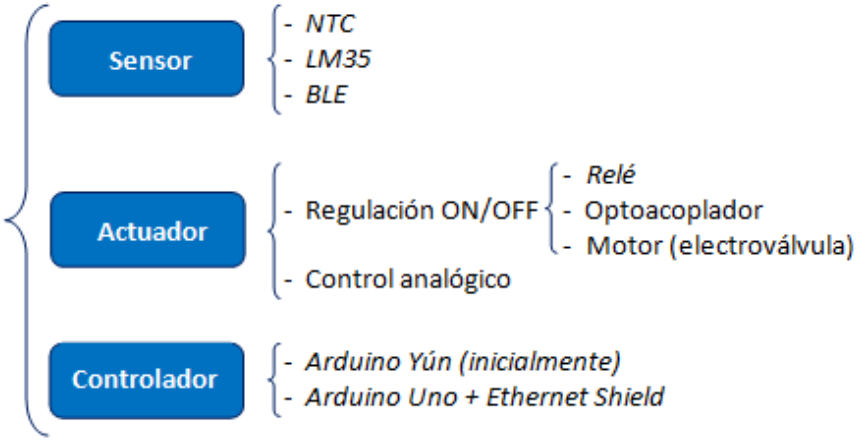


FIG. 9: MÓDULOS DEL PROTOTIPO FINAL

A continuación se realiza una pequeña explicación de los elementos fundamentales de los diferentes módulos de cada etapa y de las causas que han llevado a su elección.

2.1 SENSADO DE TEMPERATURA

Esta etapa es la encargada de determinar la temperatura ambiente de la vivienda para después compararla con la temperatura de consigna fijada por el usuario y, de este modo, determinar la acción de control necesaria.

Para esta fase, se desarrollan 3 módulos, cada uno de ellos basado en un sensor diferente: un termistor NTC, un sensor de estado sólido LM35 y un sensor BLE (Bluetooth Low Energy).

2.1.1 NTC (TERMISTOR)

Los termistores son semiconductores electrónicos con un coeficiente de temperatura que puede ser positivo o negativo. Si el coeficiente es negativo, entonces se denominan NTC (Negative Temperature Coefficient), mientras que si es positivo se denominan PTC (Positive Temperature Coefficient)

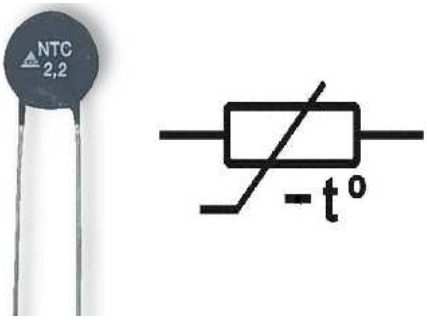


FIG. 10: NTC Y SU SÍMBOLO

El fundamento de los termistores está en la dependencia de la resistencia de los semiconductores con la temperatura, debida a la variación con ésta del número de portadores. Al aumentar la temperatura, lo hace también el número de portadores, reduciéndose la resistencia, y de ahí que presente coeficiente de temperatura negativo (NTC).

Para márgenes de temperatura reducidos, el valor de la resistencia de una NTC en función de la temperatura puede aproximarse de manera exponencial:

$$R_T = R_0 e^{\beta\left(\frac{1}{T}-\frac{1}{T_0}\right)} \qquad [1]$$

Donde,

R_T: resistencia a la temperatura absoluta T
R_0: resistencia a la temperatura absoluta de referencia T_0 (generalmente T_0=25ºC)
β: constante dentro de un intervalo reducido de temperatura.

Como se puede deducir de la expresión anterior, la respuesta de los termistores NTC ante cambios de temperatura es exponencial, tal y como se muestra en la siguiente figura.

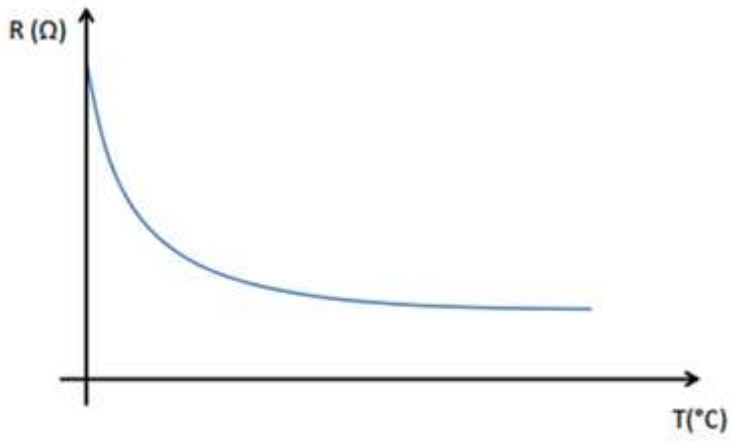


FIG. 11: CURVA R-T PARA UNA NTC

Tal y como muestra la imagen, la sensibilidad del sensor (pendiente de la curva R-T) es muy elevada a altas temperaturas, pero muy pequeña a bajas temperaturas. No obstante, este problema de no linealidad se puede solucionar mediante el uso de circuitos de linealización apropiados, obteniendo una alta sensibilidad. En consecuencia, son uno de los tipos de sensores de temperatura más empleados en termostatos.

Esta elevada sensibilidad tiene sin embargo un inconveniente: las NTCs no pueden ser empleadas para la medición de rangos de temperatura excesivamente amplios ya que las variaciones de resistencia serían demasiado grandes como para poder ser medidas con un único instrumento de medida.

Pese a esto, dado que en este proyecto la NTC se emplearía para medir la temperatura dentro de una vivienda, la cual, generalmente, no varía más de 30ºC (entre 10ºC y 40ºC), este problema de aumento excesivo de resistencia no supone ningún inconveniente.

Otra gran ventaja de las NTCs es su pequeña masa, la cual permite obtener velocidades de respuesta muy elevadas.

Así pues, tras analizar las ventajas y desventajas de los termistores, se considera que una NTC sería un tipo de sensor muy útil para este proyecto y, por tanto, se decide implementarlo en el sistema final de sensado.

2.1.2 SENSOR DE ESTADO SÓLIDO LM35

Los sensores de estado sólido son aquellos que no tienen partes móviles. Se diseñan utilizando algún fenómeno físico que cambie según la variable física que se quiera medir; en este caso la temperatura. Transforman señales no eléctricas de entrada (temperatura en este caso) en señales eléctricas de salida. Esta transformación se realiza

mediante los circuitos integrados que poseen los propios sensores de estado sólido. Existen sensores de estado sólido tanto analógicos como digitales.

Para este prototipo se emplea el sensor LM35:

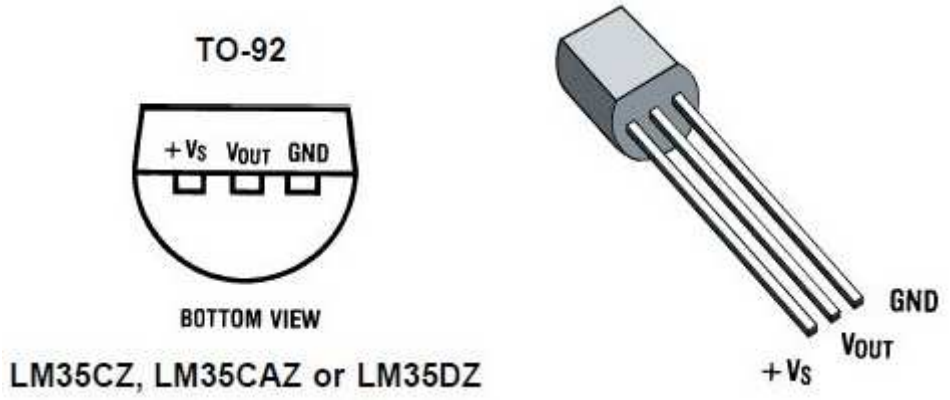


FIG. 12: SENSOR DE ESTADO SÓLIDO LM35

Su principal ventaja es su sencilla interfaz: no necesita de ningún tipo de electrónica extra para funcionar. Su mayor inconveniente es que su rango de temperatura medible está limitado hasta los 150ºC aproximadamente.

Sin embargo, como ya se ha comentado anteriormente, el rango de temperatura máximo a medir en este proyecto está en torno a los 30ºC, por lo que los sensores de estado sólido también suponen una buena opción y, por tanto, serán uno de los empleados.

2.1.3 BLE (BLUETOOTH LOW ENERGY)+ BEACONS

Bluetooth Low Energy (BLE), también llamado *Bluetooth Smart*, es una nueva tecnología inalámbrica que permite transferir información a corta distancia consumiendo poca energía. Su funcionamiento es igual que el del Bluetooth habitual: transmite información a través de ondas de radio; pero, a diferencia de éste, no envía datos de manera continua sino que lo hace intermitentemente y de manera periódica. Esto aumenta la duración de la batería de los dispositivos a la vez que reduce el coste hasta en un 80%.

La comunicación BLE suele producirse entre balizas (*Beacons*) y dispositivos móviles que posean este tipo de tecnología. Los Beacons emiten *advertisements* (pequeños paquetes de datos) a través de ondas de radio en intervalos regulares de tiempo, de modo que dispositivos dotados de BLE puedan "descubrirlos" y leer esos datos.

Una vez "descubiertos" los *Beacons*, se establece una conexión entre éstos y el dispositivo que los ha encontrado y comienza el intercambio de información. Tras la conexión, el *Beacon* ya no continúa enviando datos de manera periódica, sino que espera a recibir peticiones del dispositivo BLE conectado a él para enviar la respuesta correspondiente. Cuando el BLE conectado al *Beacon* no necesita más información de

éste, entonces se cesa la conexión y el *Beacon* queda libre de nuevo, volviendo a enviar datos periódicamente hasta ser descubierto por otro dispositivo y conectado a él.

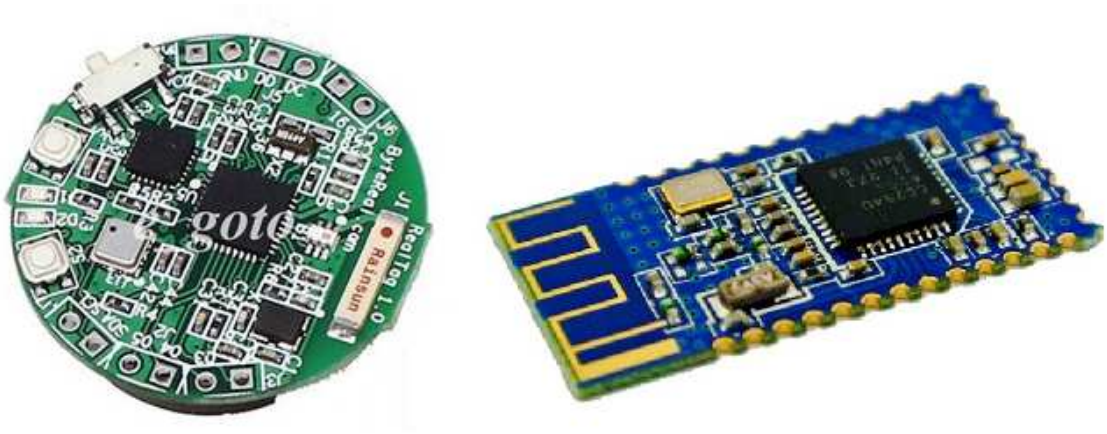


FIG. 13: BEACON Y BLE

En la actualidad, existen *Beacons* que incorporan sensores de temperatura, lo cual resulta tremendamente útil para el propósito de este proyecto. Al tratarse de sensores (balizas) inalámbricas, pueden colocarse en cualquier lugar de la vivienda sin necesidad de cables ni conexiones entre éstos y el termostato. Esto permite una mejor medición de la temperatura en hogares donde el termostato no está situado en el mejor lugar para realizar el sensado (excesivamente cerca o lejos de los radiadores, próximo a una corriente de aire, etc.). Es por esto que, como tercera opción de sensado, se seleccionó el BLE junto con *Beacons* dotados de sensores de temperatura.

2.2 ACTUACIÓN

Una vez que la temperatura de consigna es fijada por el usuario (como se explicará más adelante), y que la temperatura ambiente de la vivienda es medida a través de uno de los sensores mencionados, es momento de encender la caldera (en el caso en que eso sea necesario).

Los sistemas de actuación son aquellos que, en función del error entre la temperatura de consigna y la ambiente, encienden o apagan la caldera. Como se observa en la Fig. 9, en este prototipo se emplean dos tipos principales de actuadores: reguladores ON/OFF y reguladores analógicos.

2.2.1 REGULACIÓN ON/OFF

Se trata del tipo de regulación más antigua, más sencilla y, por tanto, más extendida. Únicamente controla el encendido o apagado de la caldera a través de distintos tipos de actuadores; pero no permite el control o modificación de la temperatura del agua que sale de la caldera hacia los radiadores.

Este proyecto se basa en los tres actuadores más empleados para este tipo de calderas: relés, optoacopladores y electroválvulas controladas mediante servomotores.

A continuación se describe el funcionamiento de cada uno de ellos.

2.2.1.1 Relé

Un relé es un interruptor accionado por un electroimán. Consta de dos partes: un circuito electromagnético (electroimán) y un circuito de contactos.

Un electroimán es una barra de hierro (núcleo) rodeada por una bobina de hilo de cobre. Al pasar corriente eléctrica por la bobina, el núcleo se magnetiza por efecto del campo magnético producido por la bobina, convirtiéndose en un imán. Cuanto mayor es la corriente que atraviesa la bobina y mayor es el número de vueltas de la bobina, mayor es la fuerza de dicho imán. Cuando se deja de aplicar corriente a la bobina, desaparece el campo magnético y el núcleo deja de actuar como un imán.

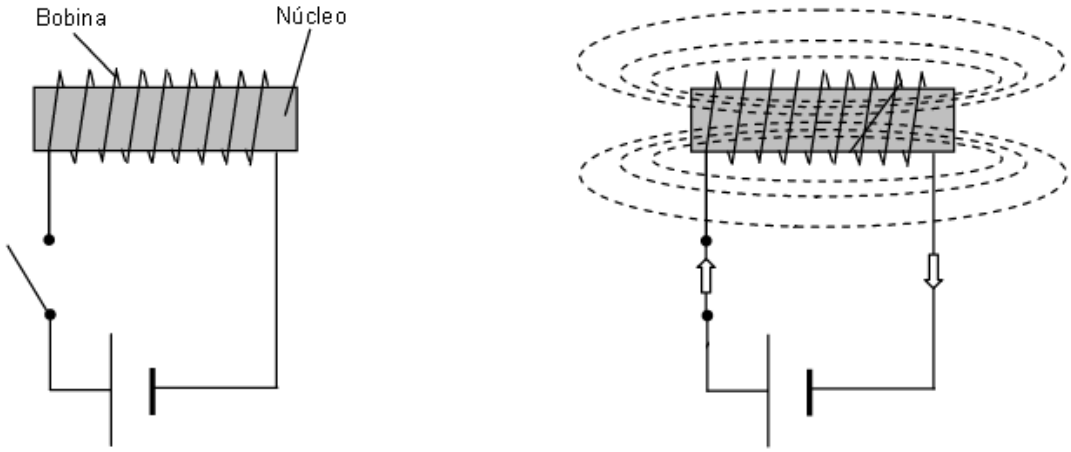


FIG. 14: ESQUEMA FUNCIONAMIENTO RELÉ

El circuito de contactos se puede observar en la siguiente figura.

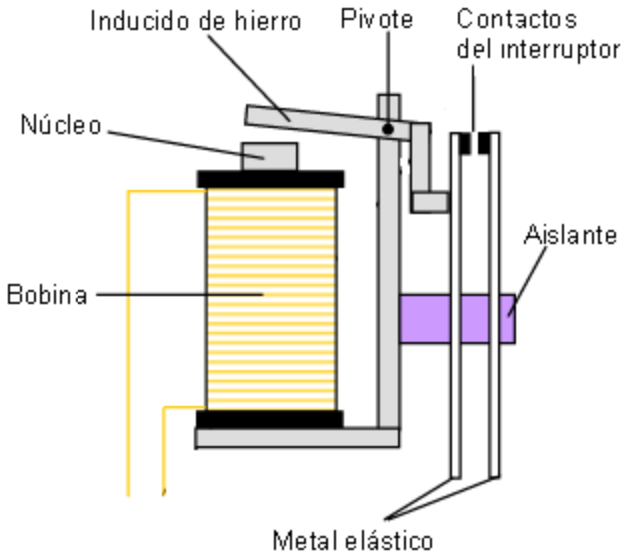


FIG. 15: ACTIVACIÓN DE CONTACTOS MEDIANTE RELÉ

Cuando no pasa corriente por la bobina, el inducido de hierro se encuentra en la posición que se muestra en la imagen y los contactos del interruptor están separados,

por lo que no pasa corriente por ellos; es decir, el circuito al que están acoplados está apagado. Sin embargo, cuando una corriente atraviesa la bobina, el campo magnético creado por el electroimán atrae al inducido de hierro hacia el núcleo, haciendo que éste empuje el contacto izquierdo del interruptor hasta que ambos contactos se tocan y, por tanto, se cierra el interruptor; dejando pasar corriente por el circuito secundario al que está conectado el relé.

Generalmente la corriente en el lado de los contactos es mucho mayor que la que circula por la bobina del relé. Ésta es una de las principales ventajas de los relés; ya que permite activar sistemas de alta tensión mediante acciones de control de baja tensión, como es el caso que nos ocupa en este proyecto. Además, como ya se ha explicado, su activación y funcionamiento es muy simple. Asimismo, tanto la instalación como el mantenimiento son sencillos y suponen un bajo coste.

Como consecuencia de todas estas características, el relé es el actuador más empleado en sistemas de calefacción debido a su bajo coste y su sencillez y rapidez de control.

La función del relé en las calderas activadas mediante este elemento es la de un simple interruptor. La caldera está conectada a una fuente de tensión y el cierre o apertura de los contactos del relé actúa como un interruptor permitiendo o no pasar la corriente y, por tanto, haciendo o no funcionar la caldera.

El esquema eléctrico de una caldera activada mediante relé quedaría del siguiente modo:

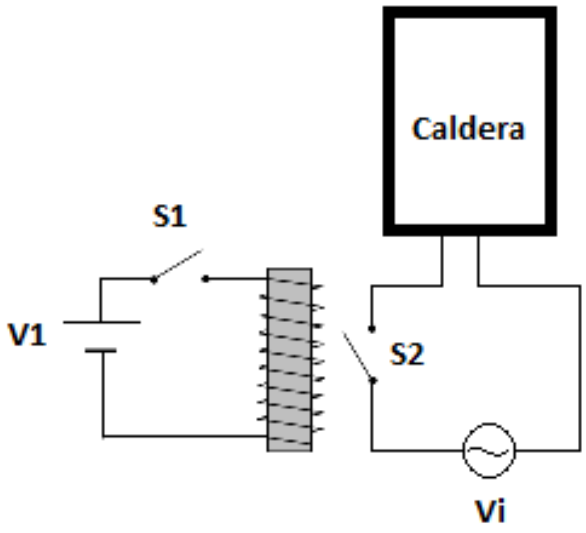


FIG. 16: ESQUEMA ELÉCTRICO DE UN SISTEMA DE CALEFACCIÓN ACTIVADO POR RELÉ

El interruptor S1 es el que controla el encendido de la caldera. Cuando S1 está cerrado, entonces pasa corriente por la bobina, creándose un campo magnético que hace cerrarse S2, conectando así la caldera con Vi. Si S1 está abierto, no llega corriente a la bobina y por tanto S2 está también abierto.

2.2.1.2 Optoacoplador

Un optoacoplador funciona de manera similar a un relé, pero evitando que el sistema que proporciona la acción de control y el sistema controlado entren en contacto. Todo optoacoplador está formado por un fotoemisor (diodo LED o diodo IRED) y un fotorreceptor (generalmente un fototransitor o un fototriac).

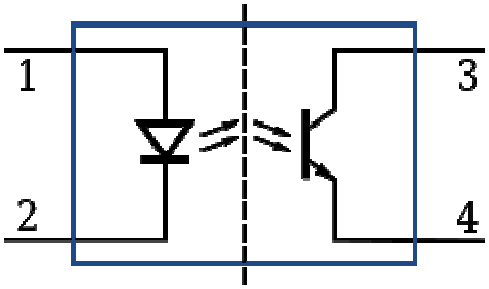


FIG. 17: ESQUEMA GENERAL DE UN OPTOACOPLADOR

Cuando la corriente en el LED/IRED (1-2) es la adecuada, éste emite luz suficiente para saturar el fototransitor, generando una corriente entre 3 y 4. Es decir, el fototransistor actúa como un interruptor cerrado cuando la corriente en el LED/IRED es suficiente para encenderlo y como interruptor abierto cuando no hay corriente en el LED/IRED o éste no tiene suficiente luminosidad (cuando la corriente no es suficiente).

La ventaja fundamental de los optoacopladores es el aislamiento eléctrico que proporcionan entre los circuitos de entrada (baja tensión) y salida (alta tensión). El único contacto entre los circuitos de entrada y salida es el haz de luz que emite el LED, lo que se traduce en una resistencia de aislamiento de miles de MΩ. Además, su control es muy sencillo ya que, como ya se ha explicado, basta con aplicar una corriente suficiente al LED para que la luminosidad de éste active el fototransistor.

A diferencia del relé, el optoacoplador no actúa como un interruptor entre la caldera y la fuente de alimentación; sino que actúa como una señal de control. La caldera se encuentra continuamente conectada a la corriente, pero sólo se activa cuando le llega la señal de control proveniente del optoacoplador. Cuando el diodo emite luz suficiente, el fototransistor entra en saturación, apareciendo una diferencia de tensión típica entre el colector y el emisor de éste, generalmente 0.2V.

El colector y el emisor del fototransistor del optoacoplador se conectan a una entrada libre de potencial de la caldera de modo que cuando el transistor está en saturación la caldera lo detecta y se enciende. Cuando el fototransistor está en corte la caldera lo entiende como un interruptor abierto y se apaga.

2.2.1.3 Válvula con servomotor o motor PAP (paso a paso)

El control de este tipo de calderas es muy sencillo. Se trata de hacer girar una válvula de modo que, cuando se desea que la caldera esté encendida, la válvula esté abierta, dejando pasar el agua; y cuando la caldera deba estar apagada, la válvula esté cerrada

evitando el paso del agua. Es la propia caldera la que se enciende si le llega agua o se apaga si deja de llegarle agua.

La apertura y cierre de la válvula se puede hacer mediante un servomotor o un motor PAP (paso a paso).

Un servomotor está formado por cuatro partes: un motor DC, un conjunto de engranajes, un circuito de control y un sensor de posición (generalmente un potenciómetro). Constan de 3 cables de conexión: alimentación, tierra y control.

FIG. 18: MICROSERVO PARA ARDUINO

El control del giro se realiza mediante señales PWM. El circuito de control interno del servomotor traduce esas señales PWM en ángulo a girar. Es la duración del pulso positivo del PWM la que determina el ángulo que se desea girar, así como su sentido. Generalmente:

- 1.5ms: 0º (el servomotor se encuentra en su posición central)
- ~2ms: 90º
- ~1ms: -90º

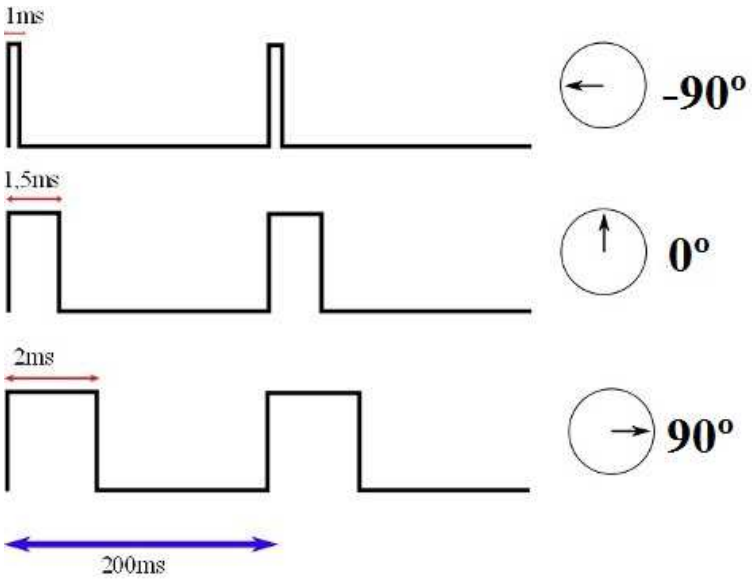


FIG. 19: CONTROL PWM DE UN SERVOMOTOR

El principal inconveniente de este tipo de motores es que, para que se mantengan en una posición determinada, es necesario enviarles continuamente el pulso PWM correspondiente. Si se deja de enviar este pulso, el motor pierde la fuerza que lo mantiene en su posición y cualquier fuerza externa, como la del agua pasando por una válvula unida a él, podría modificar el ángulo girado. Esto hace que este tipo de actuadores no resulten los más convenientes ya que absorberían corriente de Arduino continuamente. Es por esto que se plantea el uso de motores paso a paso.

Un motor paso a paso (PAP) está formado esencialmente por dos partes: un rotor sobre el que van aplicados distintos imanes permanentes y un estator en el que se encuentran un cierto número de bobinas.

El principio de funcionamiento de los motores PAP se basa en las fuerzas de atracción y repulsión ejercidas entre polos magnéticos. Se proporciona corriente alternativamente a las bobinas del estator, haciendo que los polos norte y sur de éste varíen y, con cada uno de estos cambios, debido a las fuerzas de repulsión y atracción anteriormente mencionadas, el rotor gira hasta llegar a la posición de equilibrio con las bobinas de estator.

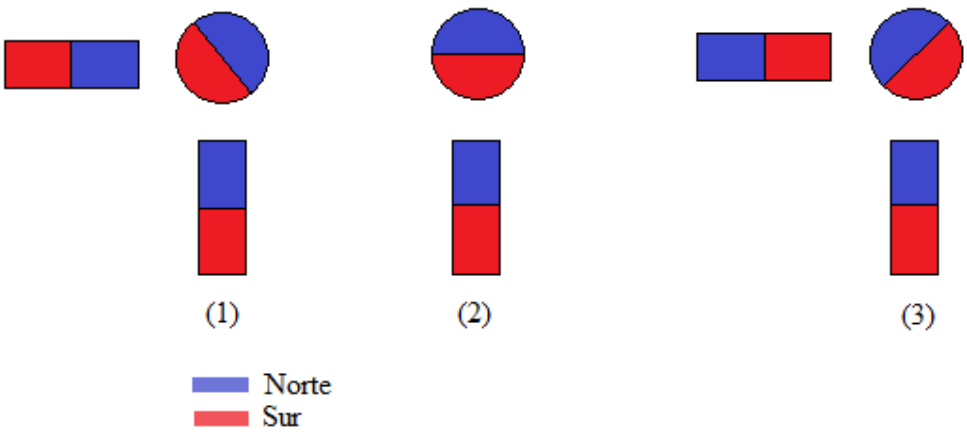


FIG. 20: FUNCIONAMIENTO MOTOR PASO A PASO

Por tanto, modificando la corriente que pasa por cada bobina se consigue el movimiento del rotor. Según el tipo de motor PAP es necesario seguir una secuencia de control u otra, como veremos más adelante.

2.2.2 CONTROL ANALÓGICO

Se trata del tipo de regulación más moderno y, por tanto, el menos empleado actualmente. Es un control más eficaz que el on-off ya que no sólo permite encender o apagar la caldera sino que, además, controla la temperatura del agua de impulsión (agua que va de la caldera a los radiadores). De este modo, cuanto mayor es la diferencia entre la temperatura de consigna y la ambiente, mayor será la temperatura del agua de impulsión, calentándose así más rápido el ambiente.

La temperatura del agua de impulsión se modifica mediante la variación de la tensión de activación de la caldera. Generalmente, este tipo de calderas se alimentan con tensiones de entre 0 y 10V, siendo 0 cuando la caldera debe estar apagada y 10 cuando se desea que el agua de impulsión esté a la mayor temperatura posible. Cuando la tensión se encuentra entre 0 y 10V el agua se calienta proporcionalmente a ésta.

Este tipo de calderas permiten un ahorro energético de hasta el 40% frente a las calderas tradicionales con activaciones on-off; ya que mediante la elevación de la temperatura del agua de impulsión se consigue que la temperatura ambiente aumente más rápidamente y, por tanto, el tiempo de encendido de la caldera se reduce notablemente.

2.3 CONTROL REMOTO

Para hacer posible el control remoto, así como para el control de todos los módulos del sistema, se emplea Arduino: una plataforma electrónica *open-source* diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares. El hardware consiste en una placa con un microcontrolador y puertos de entrada y salida tanto analógicos como digitales. El software, por su parte, consiste en un entorno de desarrollo que utiliza un lenguaje de programación basado en Processing.

Arduino fue inicialmente diseñado como una herramienta para estudiantes, ya que los microcontroladores utilizados hasta el momento de su creación eran excesivamente caros. Sin embargo, ya no sólo estudiantes, sino también programadores de todos los niveles, desde principiantes a experimentados, emplean este dispositivo para propósitos tanto caseros como profesionales. Es por esto que se selecciona Arduino como base para el presente proyecto.

Hasta el momento, Arduino ha sido utilizado en multitud de proyectos muy diversos, como pueden ser:

- Termómetros
- Domótica
- Traductores de código Morse
- Detectores de mentiras
- Piccolo: robot programable (aún en desarrollo) que puede ser programado para realizar dibujos
- Impresoras 3D
- Arduphone: teléfono móvil construido sobre un módulo Arduino
- Ardupilot: software y hardware de aeronaves no tripuladas
- Etc.

Existen multitud de placas Arduino en el mercado. De entre todas ellas, debido a las necesidades de este proyecto, se decide emplear la placa Arduino Yún, ya que es uno de los modelos más modernos y completos. Consta de 20 pines entrada/salida (7 de los cuales pueden ser empleados como salidas PWM y 6 como entradas analógicas). Posee conexión/alimentación vía USB y micro USB; así como lector de tarjetas Micro-SD. Además dispone de dos conexiones de red: una Ethernet y otra WiFi, lo cual resulta muy útil para el propósito final de este proyecto ya que permitirá conectar el dispositivo a internet y alojar en él un servidor al que poder acceder desde cualquier lugar y red, cerca o lejos del Arduino. Esta es la razón fundamental por la que se decide emplear esta placa por encima de otras opciones.

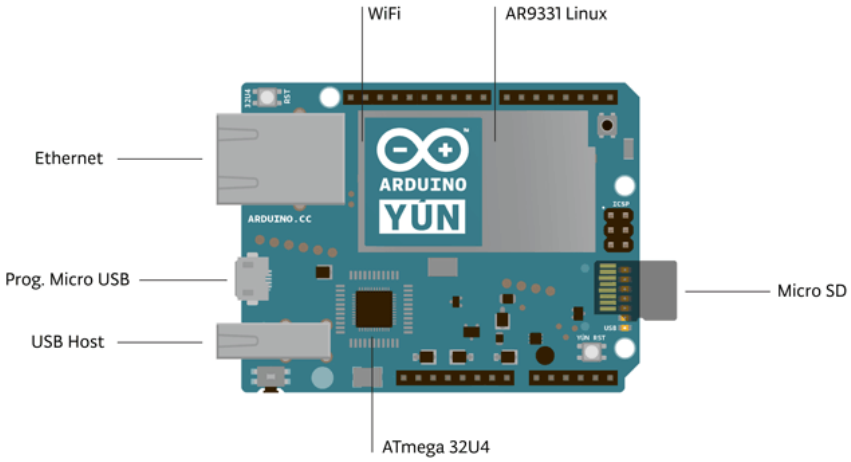


FIG. 21: ARDUINO YÚN

Este modelo es el empleado durante todo el desarrollo del proyecto. No obstante, y a pesar de que supone una muy buena solución a todas las necesidades del proyecto; ésta no es la placa empleada en el prototipo final de termostato.

Como ya se ha comentado, todo el sistema a desarrollar es modular; por lo que cada módulo es diseñado de manera independiente a lo largo de todo el proceso de desarrollo del prototipo. Sin embargo, a la hora de implementar el sistema completo (y por tanto realizar un sketch Arduino completo que incluya la programación necesaria para cada uno de los módulos) surge el siguiente problema:

Generalmente, todas las placas Arduino constan de un único microprocesador. Sin embargo, Arduino Yún dispone de dos: el ATmega 32U4, que se encarga de las funciones habituales del procesador en cualquier Arduino; y el Atheros AR9331, en el que se ejecuta Linux para las interfaces Wifi y Ethernet, y el cual también se encarga del controlador USB y de la lectura de la tarjeta SD.

Como se explicará más adelante, la comunicación entre Arduino y el BLE se realiza mediante puerto serie (se envía y se recibe información mediante una secuencia de bits). Esta comunicación, en el Arduino Yún, se realiza a través de los pines 0 y 1.

Durante el desarrollo de las diferentes etapas del proyecto, tanto la conexión serie con el BLE como la conexión wifi funcionan a la perfección. Sin embargo, surge un problema a la hora de implementar el programa final: la librería Bridge (necesaria para el funcionamiento del Wifi y el Ethernet del Yún) y la comunicación Serial empleada para la comunicación con el BLE utilizan el mismo enlace de hardware entre el ATMega y el Linux; por lo que resulta imposible emplear ambos (Wifi y BLE) al mismo tiempo. En consecuencia, se hace necesario buscar una solución que permita implementar ambas cosas a la vez.

Tras probar varias soluciones distintas (utilizar otro tipo de comunicación entre Arduino y el BLE, utilizar un puerto serie distinto al que Arduino tiene de hardware, etc), las cuales se explicarán más adelante, finalmente se decide emplear una placa Arduino Uno, la cual es más básica que el Yún, pero que aún así permite realizar todo lo necesario para el control de los distintos módulos. Sin embargo, el Uno no dispone de conexión a internet como hacía el Yún, por lo que, para poder implementar el control remoto (objetivo fundamental de este proyecto) se decide acoplar una *Shield Ethernet* al Arduino Uno de modo que este problema de conectividad quede solucionado.

Así pues, el prototipo final consta de un Arduino Uno y una *Ethernet Shield*. Arduino Uno posee un microcontrolador, 14 pines digitales de entrada/salida, 6 pines analógicos, conexión USB y de alimentación… Es muy similar al Arduino Yún, pero tienen una diferencia fundamental: Arduino Uno no dispone de conexión a internet (ni Wifi ni Ethernet). La *Ethernet shield* es una placa preparada para ser acoplada directamente sobre el Arduino Uno y dotar a éste de conexión a internet mediante un pequeño ajuste de software durante la programación del Arduino Uno.

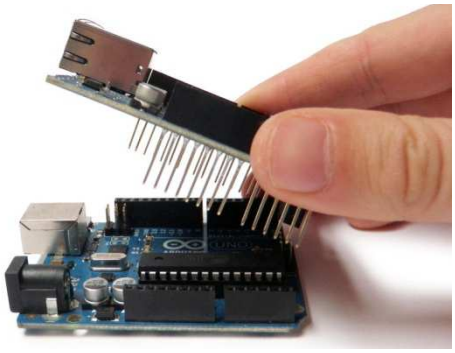


FIG. 22: ARDUINO UNO + ETHERNET SHIELD

De este modo Arduino Uno será donde se programe todo el software necesario para el correcto funcionamiento de cada uno de los módulos. Además, gracias a la conexión a internet que le proporciona la *Ethernet Shield*, será empleado como servidor web.

Para posibilitar el control remoto, se utiliza Arduino como servidor web en el que se aloja una página web desde la cual se pueden conocer distintos datos de la calefacción (estado-ON/OFF-, temperatura ambiente, temperatura actual de consigna...) y modificar la temperatura de consigna deseada.

3. DISEÑO Y DESARROLLO DEL HARDWARE

Tras la introducción, en el capítulo anterior, de los diferentes módulos, en este apartado se describe el diseño y desarrollo del hardware de cada uno de ellos. El software completo será descrito más adelante.

3.1 SISTEMAS DE SENSADO DE TEMPERATURA

Los sensores a emplear son un termistor NTC, un sensor de estado sólido LM35 y un BLE

3.1.1 NTC

Como ya se ha expuesto en el capítulo anterior, la NTC tiene muchas ventajas, pero necesita de un circuito de linealización ya que su respuesta no es lineal con la temperatura.

Para el desarrollo de este proyecto disponemos de NTCs de $R_0 = 1k$ y $R_0 = 10k$. Además, se considera que la temperatura en una vivienda se encuentra generalmente entre 10ºC y 40ºC, por lo que se decide definir un circuito que permita la linealización dentro de ese rango de temperaturas.

3.1.1.1 Linealización

Existen diversos circuitos para la linealización de una NTC. El empleado en este caso está basado en el circuito de la página 494 del libro "Operational amplifier circuits. Theory and Applications. (E. J. Kennedy)". El circuito es el siguiente:

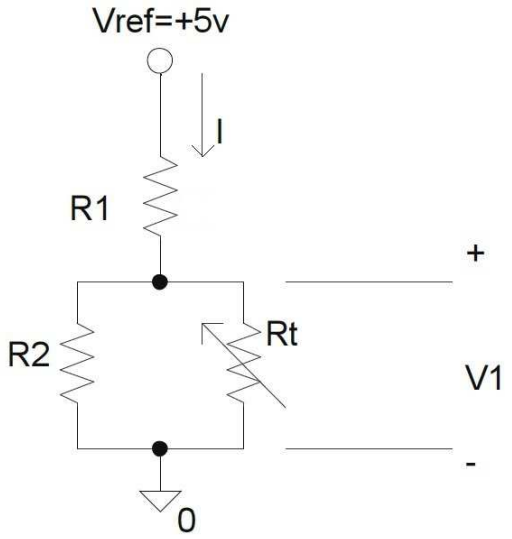


FIG. 23: CIRCUITO DE LINEALIZACIÓN PARA NTC

$$I = V_{ref}\left(R_1 + \frac{R_2 R_T}{R_2 + R_T}\right) \qquad [2]$$

$$V_1 = V_{ref}\left(\frac{1}{R_1/R_T + R_1/R_2 + 1}\right) \qquad [3]$$

Donde $R_T = R_0 e^{\beta\left(\frac{1}{T}-\frac{1}{T_0}\right)}$

Además, es sabido que si $\frac{d^2V_1}{dT^2} = 0\ \forall T \in [10ºC, 40ºC]$, entonces V_1 será totalmente lineal en ese rango de temperaturas; ya que no existe ningún máximo ni mínimo relativo en ese intervalo. Por tanto, a la hora de seleccionar los valores de R_1 y R_2, se hará de manera que:

$\frac{d^2V_1}{dT^2} \approx 0$ es decir

$$V_{ref}\left(\frac{2R_0\beta^2R_1^2R_2^3e^{\beta\left(\frac{1}{T}-\frac{1}{T_0}\right)+\frac{\beta}{T_0}}}{T^4\left(R_0R_1e^{\frac{\beta}{T}} + R_0R_2e^{\frac{\beta}{T}} + R_1R_2e^{\frac{\beta}{T_0}}\right)^3} + \frac{R_0\beta R_1R_2^2(2T-\beta)e^{\beta\left(\frac{1}{T}-\frac{1}{T_0}\right)}}{T^4\left(R_0R_1e^{\frac{\beta}{T}} + R_0R_2e^{\frac{\beta}{T}} + R_1R_2e^{\frac{\beta}{T_0}}\right)^2}\right) \approx 0 \qquad [4]$$

Puesto que no sería eficaz comprobar el cumplimiento de esta ecuación para todos los valores del intervalo [10ºC,40ºC] se decide comprobarlo para $T_L = \frac{40+10}{2} = 25$ ya que es el punto medio del intervalo y cerca del cual nos encontraremos generalmente dentro de una vivienda.

Por otro lado, los valores de las resistencias son escogidos también de modo que las corrientes absorbidas por el circuito de linealización y por la NTC sean pequeñas, por dos motivos:

- Minimizar la corriente absorbida de la fuente (Arduino), ya que éste no puede proporcionar más de 40mA/pin.
- Minimizar la corriente que circula por la NTC para reducir así su autocalentamiento, el cual daría lugar a errores de medición.

A partir de las ecuaciones [2], [3] y [4], mediante una hoja Excel (Anexos 1 y 2), se calcula el valor de las resistencias R_1 y R_2 necesarias para los circuitos correspondientes a cada una de las NTCs (1k y 10k) y de la correspondiente ecuación linealizada de V_1. Esto se lleva a cabo del siguiente modo:

- Se selecciona un valor de R_1 y, con la ecuación [4] se calcula el valor de R_2 correspondiente.
- Mediante la ecuación [1] se calcula el valor de R_T para temperaturas de entre 10ºC y 40ºC.

- Se determina el valor de V_1 a través de la ecuación [3] para temperaturas entre 10ºC y 40ºC.
- Una vez obtenidos todos estos valores, se representa en una gráfica V_1 frente a T y se obtiene la ecuación lineal que más se aproxima a esa gráfica. Dicha ecuación será empleada más adelante cuando se cree el software de Arduino para la lectura de la temperatura.

Todo este proceso se puede observar en las tablas de los anexos 1 y 2; donde cada columna representa lo siguiente:

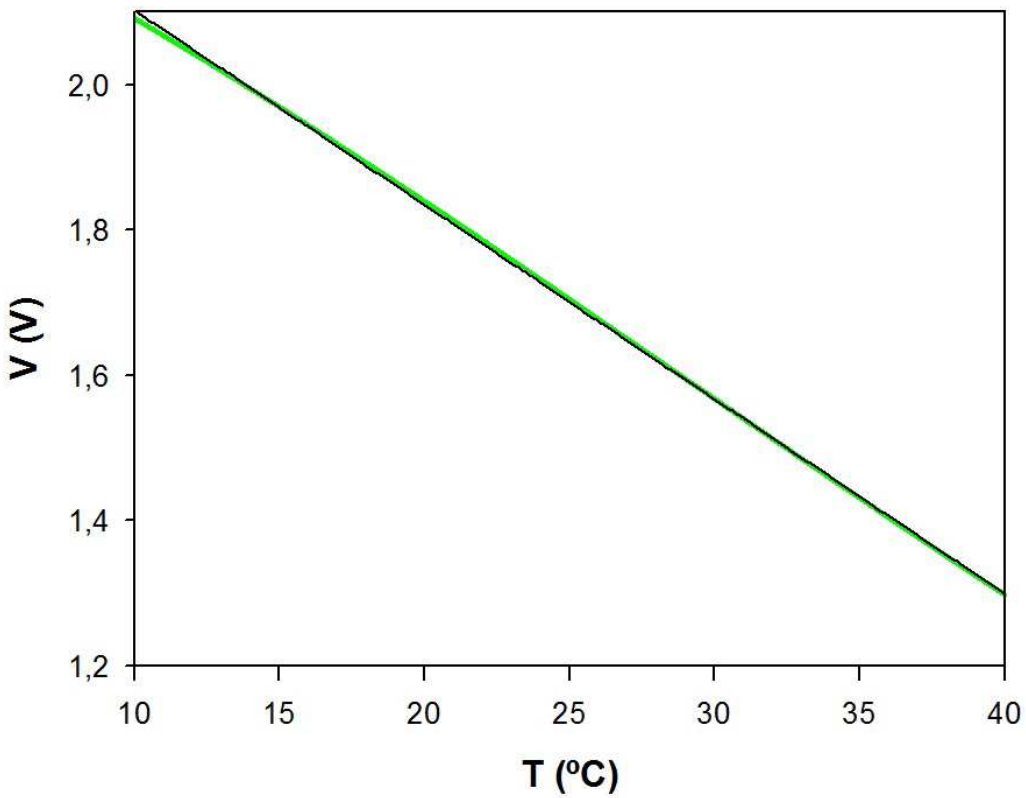
- T: temperatura [ºC]
- R_t: resistencia de la NTC a temperatura T
- V_1: valor de la tensión V_1 obtenido mediante la ecuación [3]
- Aproximación: valor de la tensión V_1 obtenido mediante la ecuación lineal obtenida
- I_1: corriente total absorbida por el circuito de linealización
- I_{NTC}: corriente absorbida por la NTC

Veamos pues los valores obtenidos para cada NTC:

- **NTC 1k:**

  Se determinan los siguientes valores: R_1 = 995 Ω y R_2 = 1k616Ω

  A partir de los valores obtenidos en la tabla del anexo 1, se tiene la siguiente gráfica:



GRÁFICA 1: LINEALIZACIÓN DE V1 PARA NTC DE 1K

La ecuación aproximada de V_1 (linealización) es:

$$V_1 = -0.02677 \cdot T + 2.37027 \qquad \textbf{[5]}$$

Donde T está expresada en ºC.

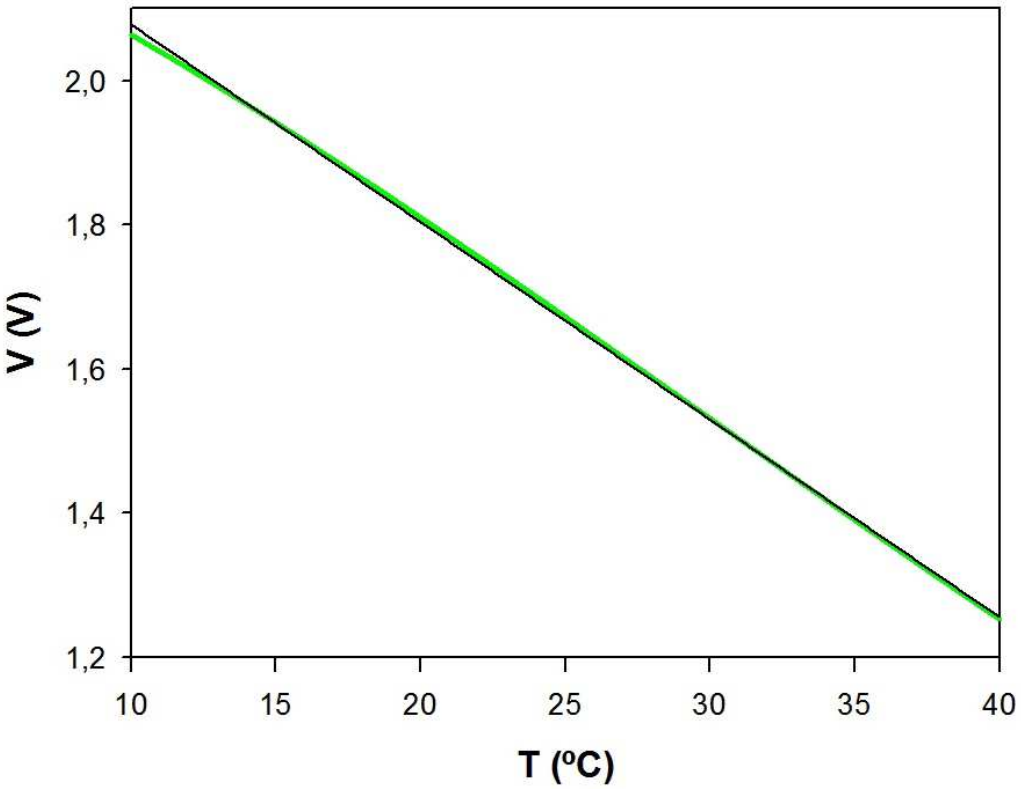
Y el error relativo en el rango de temperaturas deseado es muy pequeño, por lo que la aproximación es bastante exacta.

Además, la corriente total absorbida por el circuito es muy pequeña (del orden de 2-3mA); así como la corriente que circula por la NTC (alrededor de 1-2mA).

- **NTC 10k**

  Se determinan los siguientes valores: R1 = 9k95 Ω y R2 = 14k94Ω

  A partir de los valores obtenidos en la tabla del apéndice 2, se obtiene la siguiente gráfica:



GRÁFICA 2: LINEALIZACIÓN DE V1 PARA NTC DE 10K

Como se observa en la gráfica, la ecuación aproximada de V_1 resulta:

$$V_1 = -0.02741 \cdot T + 2.35268 \qquad \textbf{[6]}$$

Y el error relativo en el rango de temperaturas deseado es muy pequeño, por lo que la aproximación es bastante exacta.

Además, la corriente total absorbida por el circuito es muy pequeña (del orden de 0.2-0.3mA); así como la corriente que circula por la NTC (alrededor de 0.1-0.2mA).

3.1.1.2 Amplificación

En el circuito de la Fig. 23, conforme aumenta la temperatura, R_t disminuye, haciendo así también disminuir V_1, como se puede deducir de la ecuación [2]. Tal y como se observa en las tablas de los anexos 1 y 2, la tensión de salida V_1 de los circuitos de linealización varía entre 1.25V y 2.10V aproximadamente en ambos casos (R_0 = 1k y R_0 = 10k). Es decir, la máxima variación de tensión que se producirá en el rango de temperaturas seleccionado (10ºC-40ºC) será de menos de 1V.

Teniendo en cuenta que las entradas analógicas de Arduino soportan tensiones de hasta 5V, lo ideal sería amplificar esa variación para que se encuentre entre 0 y 5V aproximadamente, obteniendo así una resolución mayor a la hora de realizar la medición de la temperatura a través de Arduino. Para llevar a cabo dicha amplificación se utiliza un amplificador de instrumentación (INA126P).

Como ya se ha comentado, se desea que la tensión de salida del AI (amplificador de instrumentación) y, por tanto, la de entrada a Arduino, se encuentre aproximadamente entre 0 y 5V; sin embargo, eso es en principio imposible, ya que la tensión V_1 mínima es de alrededor de 1.25. Para solucionar esto, se diseña el sistema de amplificación de modo que la tensión a la entrada del AI sea aproximadamente 0V cuando V_1 es mínima. Así pues, el sistema de amplificación queda del siguiente modo:

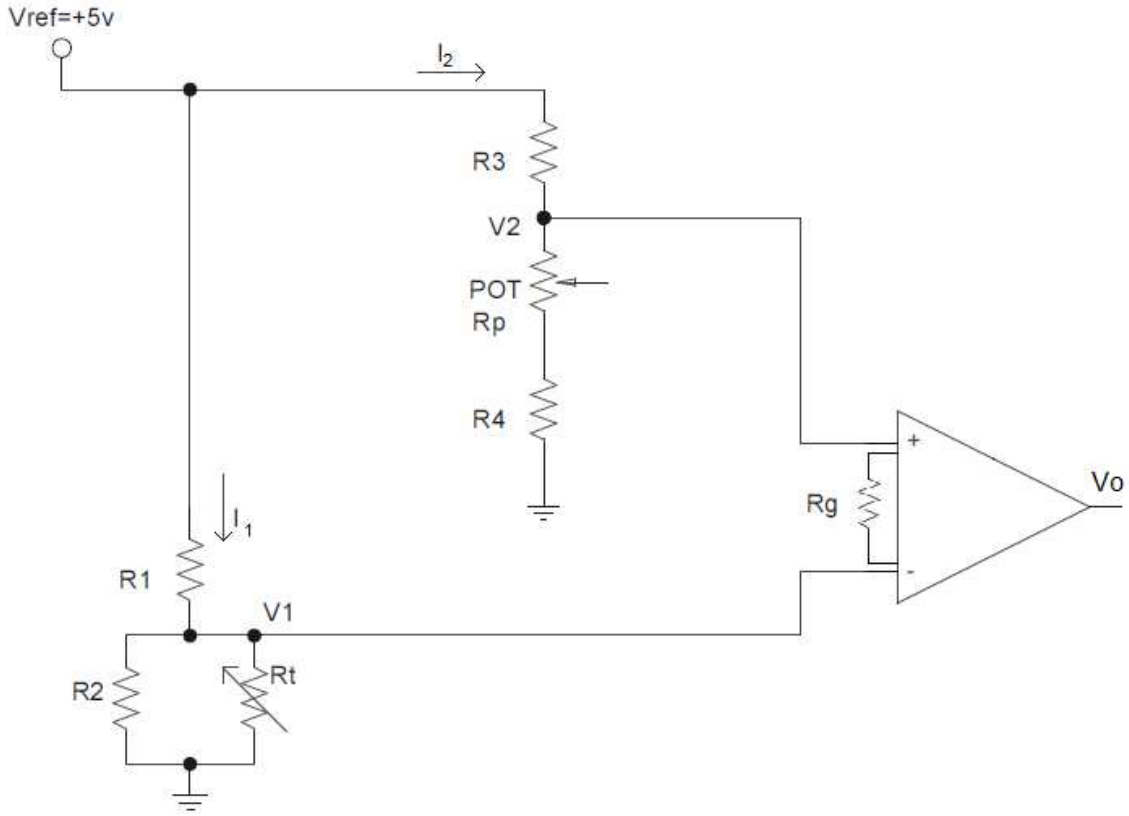


FIG. 24: LINEALIZACIÓN Y AMPLIFICACIÓN PARA NTC

Donde:

$$I_2 = V_{ref}\frac{1}{R_3 + R_p + R_4} \qquad [7]$$

$$V_2 = V_{ref}\left(1 - \frac{R_3}{R_3 + R_p + R_4}\right) \qquad [8]$$

Se utiliza un potenciómetro (R_p) en lugar de una resistencia de valor fijo ya que de este modo el ajuste que se obtiene es más fino; y más aún gracias al empleo de un potenciómetro multivuelta.

La tensión a amplificar en este circuito es:

$$V_i = V_2 - V_1 \qquad [9]$$

Y la tensión de salida:

$$V_0 = G \cdot V_i = G(V_2 - V_1) \qquad [10]$$

Para la amplificación se emplea un amplificador de instrumentación INA126P, cuya ganancia se puede expresar del siguiente modo:

$$G = 5 + \frac{80k}{R_g} \qquad [11]$$

Con este sistema, lo que se pretende es hacer que la tensión a amplificar (V_i=V_2-V_1) sea 0V cuando la temperatura sea la mínima a medir (10ºC). Además, igual que en el apartado anterior, se desea que la corriente absorbida por esta parte del circuito también sea pequeña, para no demandar demasiada corriente al Arduino que lo alimenta ya que éste sólo puede proporcionar un máximo de 40mA

Veamos pues cuales son los valores obtenidos para cada NTC:

- **NTC 1k:**

  Se desea que Vi ≈ 0 cuando T = 10ºC, por lo que, dado que

  V_1(10ºC) = 2.09V (anexo 1)

  Entonces se intenta que V_2 = 2.09V. Para ello se determina que los valores de resistencias necesarios son:

  R_3 = 1k778Ω

  R_p = 635Ω

  R_4 = 995Ω

  Obteniéndose:

  V_2 = 2.10 ≈ 2.09V I_2 = 1.3mA

  (V_2 = 2.12V si se calcula de manera teórica con la ecuación [6]; sin embargo, el valor real medido en el circuito es de 2.10V)

Ahora sólo queda calcular la ganancia necesaria para obtener la máxima resolución posible y la resistencia Rg necesaria para ello.

$$V_{i.mín} = V_2 - V_{1.máx} \qquad \textbf{[12]}$$

$$V_{i.máx} = V_2 - V_{1.mín} \qquad \textbf{[13]}$$

Así

$V_{i\ mín}$ = 2.10-2.09 = 0.01V y $V_{i\ máx}$ = 2.10-1.30 = 0.8V

Se desea que

$V_{o\ máx}$ = 5V,

por lo que con la ecuación [10] se determina

$G \leq 6.25$

Así pues, por la ecuación [11],

$Rg \geq 64k\Omega$

Puesto que 64kΩ no es un valor normalizado de resistencia, se escoge

Rg = 64k9Ω

obteniéndose una ganancia final, mediante la ecuación [11] de

G = 6.23

Con los anteriores valores y con la ecuación [10]

$V_{o\ mín}$ = 0.06V y $V_{o\ máx}$ = 4.98V

y teniendo en cuenta que

$$r = \frac{V_{o.máx} - V_{o.mín}}{T_{máx} - T_{mín}} \qquad \textbf{[14]}$$

entonces, la resolución obtenida es

r = 164mV/ºC

- **NTC 10k:**

Se desea que Vi ≈ 0 cuando T = 10ºC, por lo que, dado que

V_1(10ºC) = 2.06V (anexo 2)

entonces se intenta que V_2 = 2.06V. Para ello se determina que los valores de resistencias necesarios son

R_3 = 1k874Ω

R_p = 629Ω

R_4 = 1kΩ

Obteniéndose

V_2 = 2.07V ≈ 2.06V I_2 = 1.27mA.

Ahora sólo queda calcular la ganancia necesaria para obtener la máxima resolución posible y la resistencia Rg necesaria para ello.

A partir de las ecuaciones [12] y [13]

$V_{i\ mín} = 2.07-2.06=0.01V$ y $V_{i\ máx} = 2.07-1.25=0.82V$

Se desea que

$V_{o\ mín} = 5V$

por lo que con la ecuación [10] se determina

$G \leq 6.09$

Así pues, por la ecuación [11]

$Rg \geq 73k39\Omega$

Puesto que 73k39Ω no es un valor normalizado de resistencia, se escoge

$Rg = 75k\Omega$

obteniéndose una ganancia final, mediante la ecuación [11] de

$G = 6.07$

Con los anteriores valores y con la ecuación [10]

$V_{o\ mín} = 0.06V$ y $V_{o\ máx} = 4.98V$

Entonces, mediante la ecuación [14]

$r = 164mV/ºC$

3.1.1.3 Comparación

Ya están pues definidos todos los valores de las resistencias para los circuitos de ambas NTCs. Queda por tanto decidir cuál de las dos NTCs disponible proporciona mejores resultados. Como se ha observado, la resolución obtenida con ambas NTCs es la misma, por lo que esto no sirve como factor decisivo. Por ello, se realizan dos pruebas:

Primero, se colocan ambos sensores en un entorno controlado y se realiza la medición de la temperatura desde el momento en el que se enciende la calefacción hasta tiempo después de apagarse ésta. A continuación se pueden observar los resultados obtenidos.

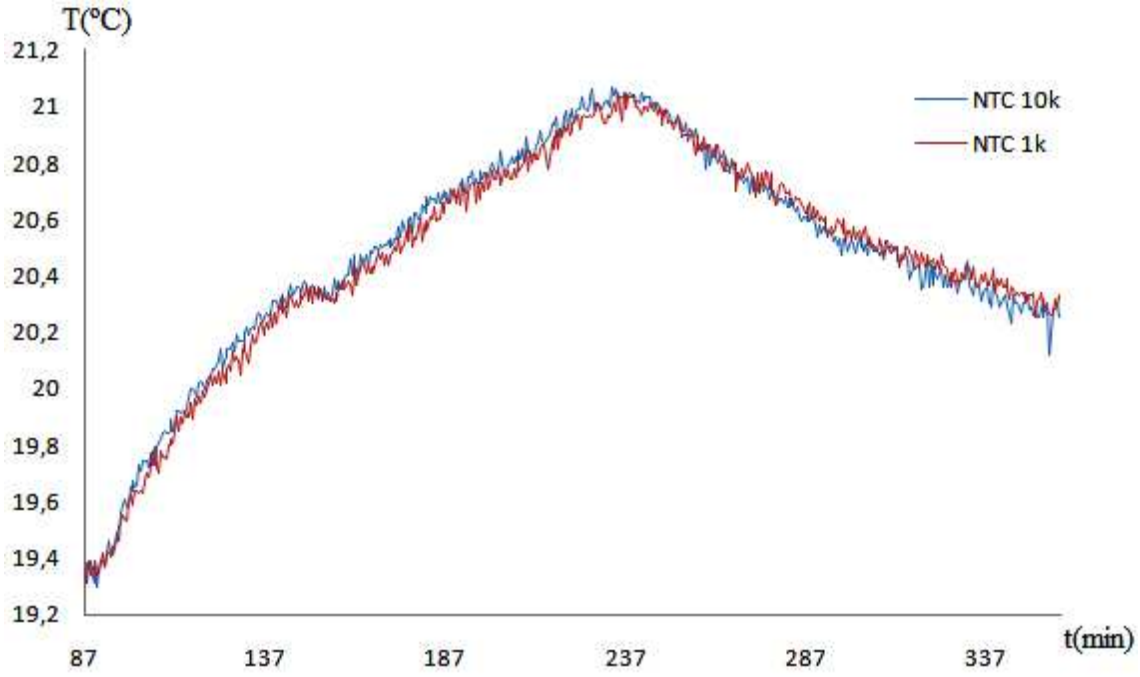


FIG. 25: COMPARATIVA RESULTADOS NTCS 1K Y 10K ANTE ENCENDIDO Y APAGADO DE CALEFACCIÓN

Para la creación de esta gráfica se toman valores de temperatura cada 30s; lo cual explica las fluctuaciones que se pueden observar en la imagen. Sin embargo, analizando la figura, es difícil llegar a conclusiones definitivas; ya que ambas NTCs proporcionan prácticamente el mismo valor de temperatura y tienen variaciones similares. Si bien puede que parezca que las fluctuaciones de los datos de la NTC de 10k son menores cuando la temperatura se encuentra en aumento, ocurre lo contrario cuando ésta disminuye después de apagarse la caldera. Por tanto, resulta imposible decidir qué NTC emplear a fin de obtener los resultados más fiables.

En consecuencia, se decide realizar una segunda prueba. Ésta vez se colocan los sensores en una zona cercana a una puerta, cuya apertura supone una corriente de aire que enfría el ambiente. Primero se espera a que la temperatura medida por los sensores se estabilice y, a continuación, se abre la puerta para comprobar el efecto de la corriente de aire sobre las mediciones. Los resultados obtenidos son los siguientes:

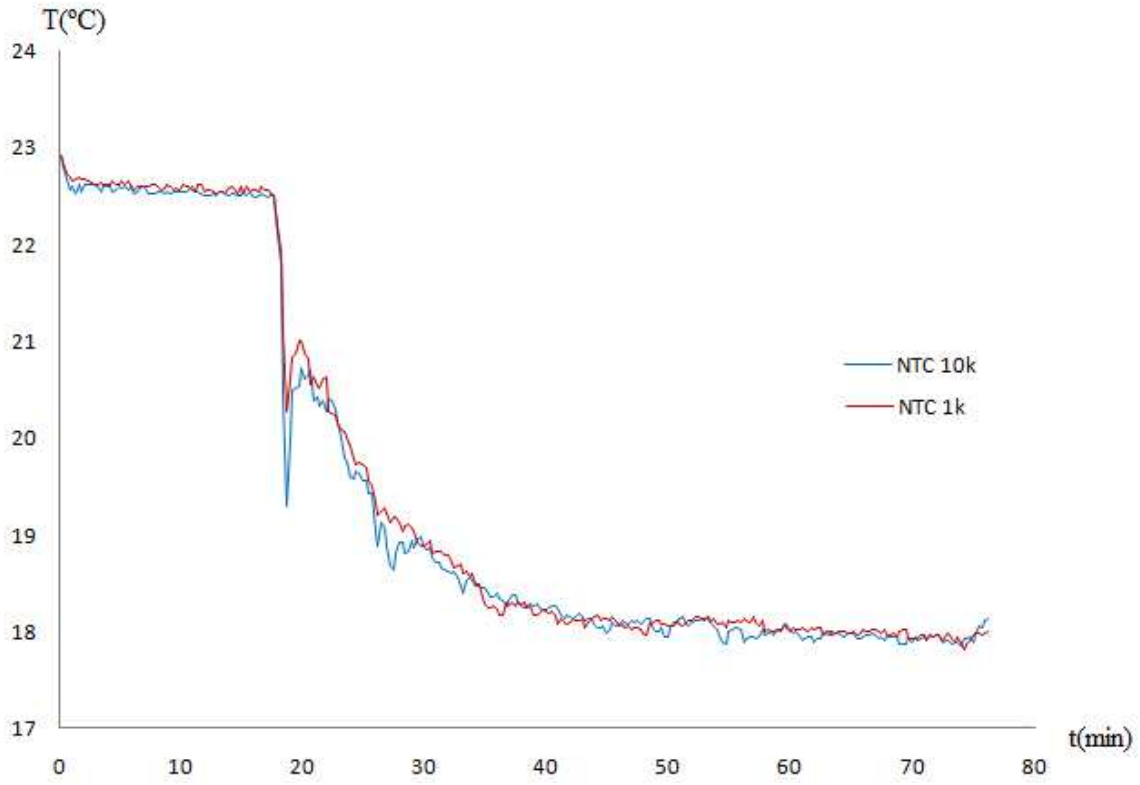


FIG. 26: COMPARATIVA RESULTADOS NTCS 1K Y 10K ANTE CORRIENTE DE AIRE

Tras el análisis de esta gráfica se decide emplear la NTC de 1k por las siguientes razones:

- Las fluctuaciones de las medidas cuando la temperatura está estabilizada (~22.7ºC y 18ºC) son menores cuando se emplea la NTC de 1k
- Como se observa en la figura, cuando comienza el cambio brusco de temperatura (corriente de aire) ambas NTCs cometen cierto error en la medición: la temperatura medida disminuye bruscamente para luego aumentar ligeramente antes de comenzar a disminuir de manera más o menos continua. Sin embargo, ese error es mayor cuando se emplea la NTC de 10k, ya que la disminución inicial es casi 1ºC mayor que la que ocurre al utilizar la de 1k.

Así pues, se decide definitivamente emplear la NTC de 1k para este proyecto. Una vez tomada esta decisión, sólo queda diseñar la placa impresa del circuito completo. Sin embargo, antes de esto es necesario incorporar un nuevo elemento al circuito:

El AI empleado en este circuito es el INA126P, el cual requiere un voltaje de alimentación de ±5V. Arduino proporciona +5V, por lo que se hace necesaria la utilización de un conversor DC-DC que convierta esa tensión de +5V a ±5V. Para este cometido se selecciona el conversor TRA 1-0521 de Traco Power.

De este modo, el circuito final de sensado mediante NTC queda:

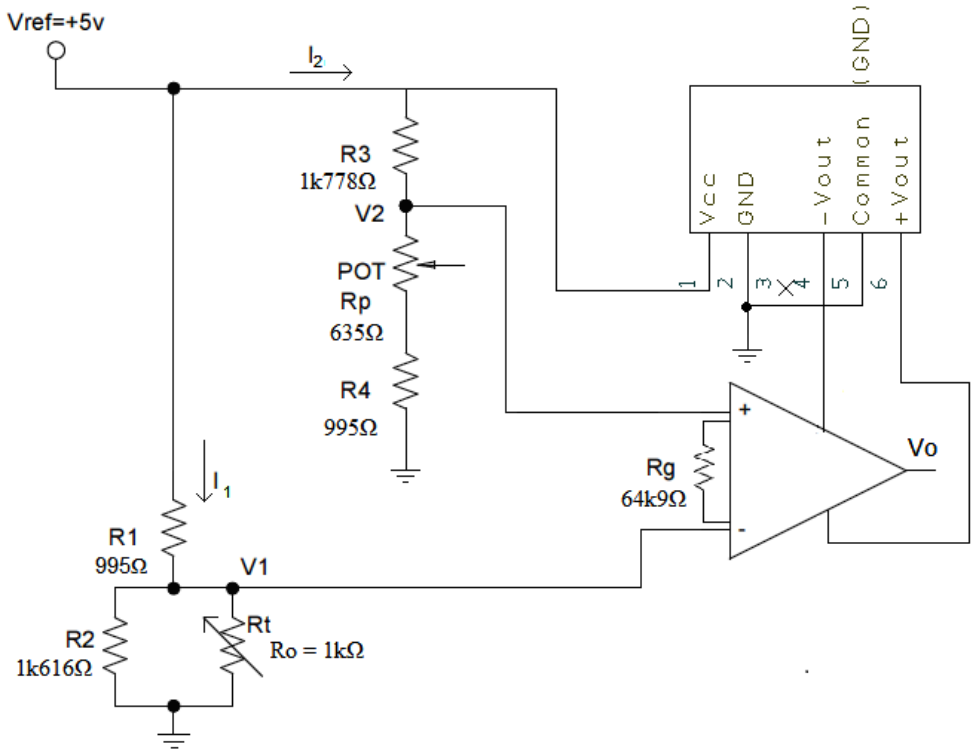


FIG. 27: CIRCUITO FINAL SENSADO NTC

Una vez configurado todo el circuito, ya solo queda diseñar una PCB (Printed Circuit Board) que haga más sencilla la utilización de éste:

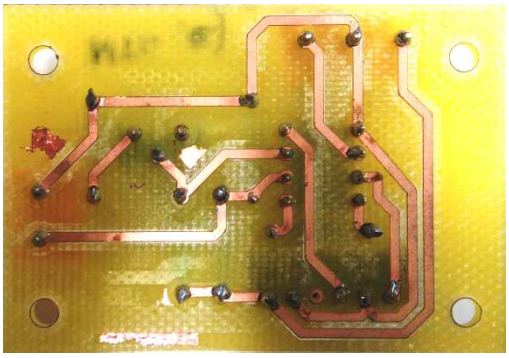


FIG. 28: PARTE IMPRESA DE LA PCB PARA EL MÓDULO DE SENSADO MEDIANTE NTC

FIG. 29: VISTA SUPERIOR DE LA PCB PARA EL MÓDULO DE SENSADO MEDIANTE NTC

3.1.2 SENSOR DE ESTADO SÓLIDO LM35

Este tipo de sensores, como ya se ha comentado en aparatados anteriores, son muy sencillos de utilizar. En este caso, se decide emplear el sensor LM35 debido a las siguientes características:

- Está calibrado directamente en ºC: es decir, la tensión de salida del sensor es directamente proporcional a la temperatura en ºC. Esta es una gran ventaja de este sensor ya que, de este modo, no se necesita ningún tipo de circuito de linealización ni es necesario realizar conversiones de unidades (ºF-ºC)
- Opera entre 4 y 30V, lo que resulta perfecto, ya que en este caso ha de ser alimentado a través del Arduino, el cual proporciona 5V
- Consume menos de 60µA
- Tiene un bajo autocalentamiento

El conexionado de este sensor es muy sencillo:

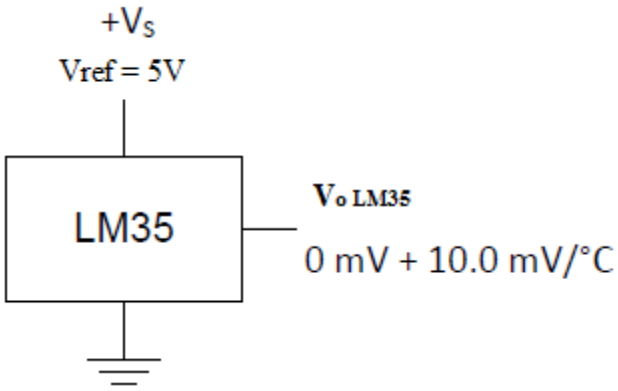


FIG. 30: CONEXIONADO BÁSICO LM35

El sensor LM35 tiene una tensión de salida

$$V_{0.LM35} = 10mV/^{o}C = 0.01 \cdot T \qquad \textbf{[15]}$$

La tensión máxima de salida del LM35, cuando T = 40ºC, será, por la ecuación [15]

$$V_{o\ LM35\ máx} = 40 \cdot 10 = 400mV.$$

La máxima tensión que admiten las entradas analógicas de Arduino (entradas a las que irá conectado el sensor) es de 5V; por lo que, igual que ocurría con las NTCs, lo ideal sería qye el LM35 proporcionara una tensión de entre 0 y 5V, para obtener una mayor resolución. Para este fin, se utiliza de nuevo el amplificador INA126P y, al igual que en el caso de las NTCs, es necesario incorporar al circuito un conversor DC-DC para poder alimentar el AI a través del Arduino.

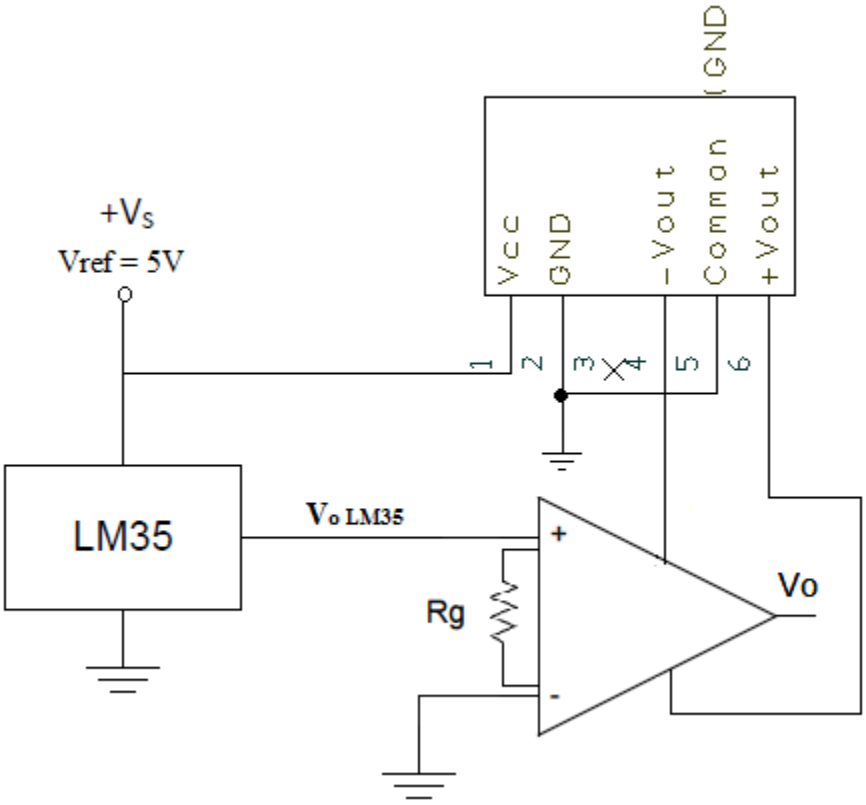


FIG. 31: CIRCUITO FINAL DE SENSADO Y AMPLIFICACIÓN CON LM35

Donde

$$V_O = G \cdot V_{O.LM35} = G \cdot 0.01 \cdot T \qquad \textbf{[16]}$$

Para que la tensión máxima sea Vo máx = 5V, entonces, teniendo en cuenta que $V_{o\ LM35\ máx}$ = 400mV y mediante la ecuación [10]

$G \leq 12.5$

Así, por la ecuación [11]

$Rg \geq 10K67\Omega$

Con el fin de utilizar una resistencia normalizada, se selecciona

$Rg = 11k\Omega$

obteniéndose una ganancia teórica, mediante la ecuación [11] de

$G = 12.27$

y por tanto, por la ecuación [10]

$V_{o\ máx} = 12.27 \cdot 0.4 = 4.91V$

Así pues, a través de la ecuación [14]

$r = 123mV/ºC.$

Tras implementar el circuito y comprobar su correcto funcionamiento, se diseña la PCB correspondiente.

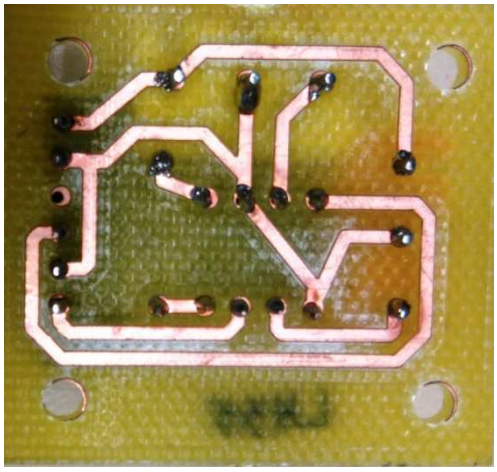


FIG. 32: PARTE IMPRESA DE LA PCB DEL MÓDULO PARA EL SENSADO MEDIANTE LM35

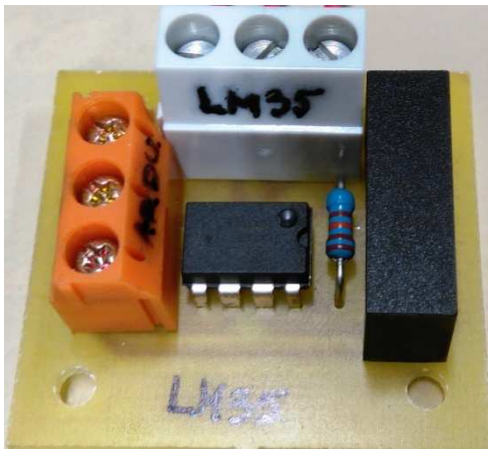


FIG. 33: VISTA SUPERIOR DE LA PCB DEL MÓDULO PARA EL SENSADO MEDIANTE LM35

3.1.3 BLE (BLUETOOTH LOW ENERGY) Y BEACONS

Este sensor supone una gran ventaja respecto a los anteriormente utilizados ya que, al comunicarse mediante Bluetooth Low Energy, no necesita cableado y, por tanto, no necesariamente debe situarse junto al termostato. De este modo, en hogares donde la ubicación del termostato no es óptima para la medida de temperatura (habitaciones más frías o más cálidas que el resto, corrientes de aire, demasiado cerca o lejos de radiadores, etc), este sensor permite realizar las mediciones en el lugar más apropiado. El chip BLE es el que se encuentra dentro del termostato cableado al Arduino y el Beacon se coloca en el lugar deseado, produciéndose la comunicación Bluetooth entre ambos.

Para la implementación de este tipo de sensores en el presente proyecto se emplea una baliza (*Beacon*) Realtag 1.0 y un módulo *Bluetooth Low Energy* (BLE) conectado al Arduino y controlado a través de éste.

Los módulos BLE son generalmente controlados mediante programas diseñados específicamente para ese fin, como puede ser el programa BTool, cuya interfaz se puede observar en la siguiente imagen.

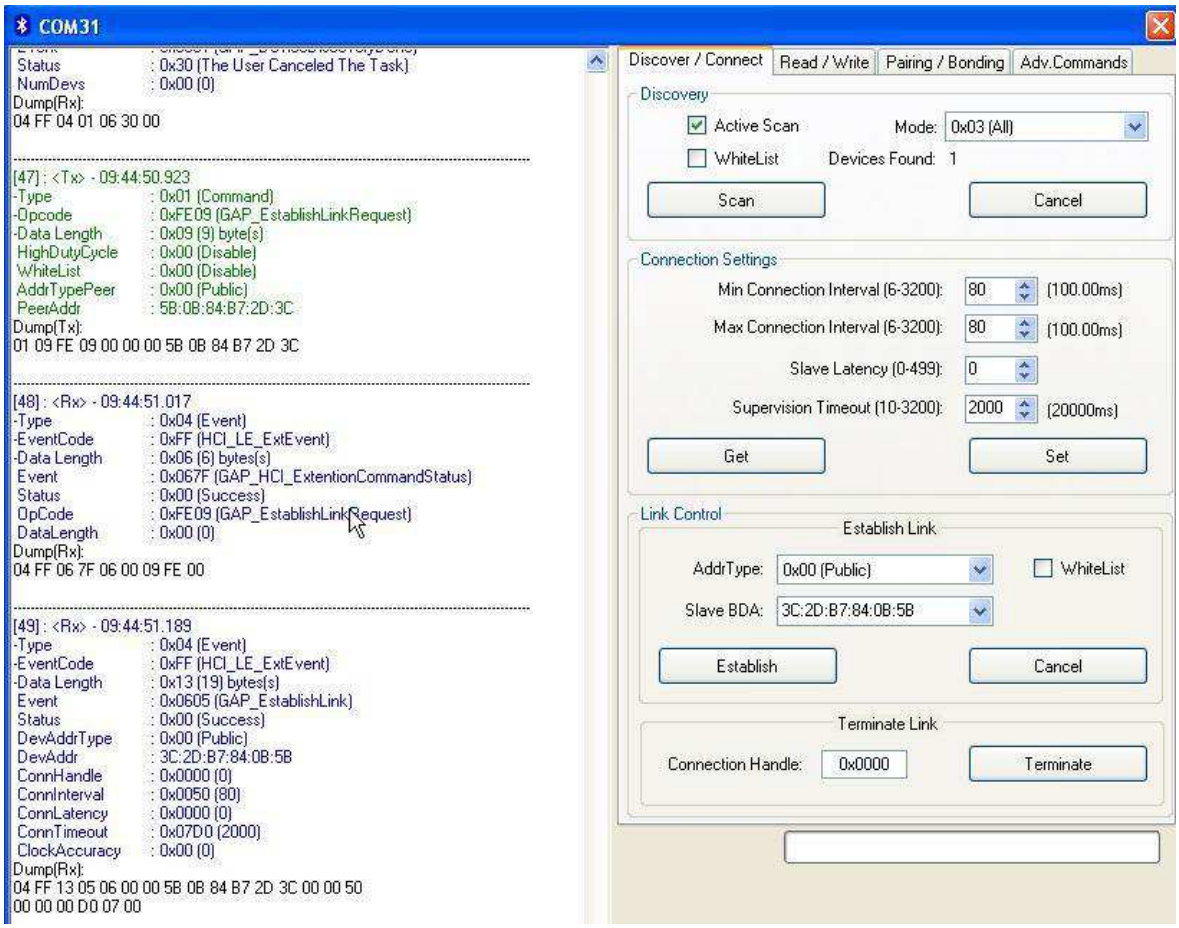


FIG. 34: INTERFAZ BTOOL

En la imagen superior, la parte de la derecha es la que permite al usuario seleccionar las acciones que desea realizar desde el BLE (buscar *Beacons*, conectarse a un *Beacon*, solicitar el dato de temperatura, etc). La mitad izquierda de la imagen muestra, en verde, los comandos enviados al BLE para realización de dichas acciones; y en azul, los comandos de respuesta del *Beacon*.

No obstante, BTool necesita de un ordenador para ser manejado; por lo que su utilización en este proyecto resulta imposible, ya que el prototipo final del termostato no dispondrá de un ordenador, sino que será controlado íntegramente por el Arduino. Es por esto que se hace necesario un programa de Arduino que permita enviar los comandos necesarios en cada momento; así como recibir e interpretar los recibidos.

Antes de realizar el programa de Arduino que permita obtener la temperatura de los *Beacons* a través del módulo BLE, se necesita conocer la secuencia de comandos que se debe enviar al módulo BLE para la adquisición de ese dato; así como los comandos de respuesta que se obtendrán, para poder después interpretarlos correctamente. Para ello, es necesario conectar el BLE al ordenador y obtener la temperatura utilizando las

funciones de BTool para así poder observar en pantalla los distintos comandos que el programa envía y recibe a lo largo de todo el proceso.

Sin embargo, como se puede ver en la Fig. 13, el módulo BLE no dispone de conexión USB que facilite conectarlo al ordenador. Sí dispone, no obstante, de pines que permiten la conexión serie con otros dispositivos: un pin (R_x) que recibe datos o comandos y otro (T_x) que los envía o transmite. Arduino dispone también de este tipo de conexión: posee pines R_x y T_x, los cuales, generalmente, están conectados con el puerto USB de la placa. Para poder conectar inicialmente el BLE al ordenador mediante puerto serie y controlarlo mediante BTool se emplea un Arduino Uno ya que en esta placa el microprocesador puede ser extraído y, por tanto, se puede emplear simplemente como un hardware para la conexión entre un puerto serie (BLE) y uno USB (ordenador).

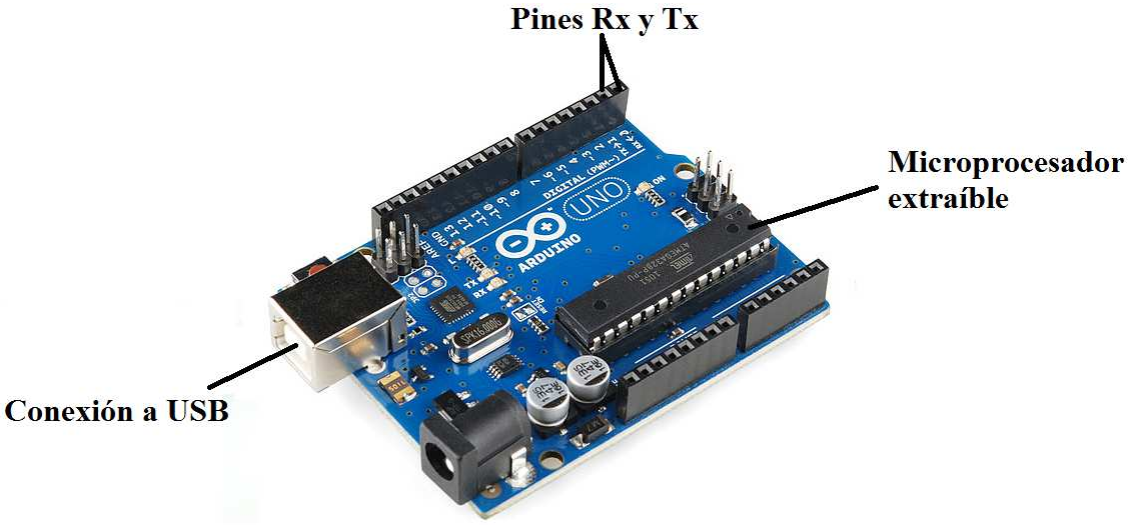


FIG. 35: CONEXIONES SERIE ARDUINO UNO

En la tabla del anexo 3 se muestra la secuencia de comandos (enviados y recibidos) obtenida en BTool para la lectura de la temperatura del sensor del Beacon, desde la inicialización del BLE hasta la lectura del comando de temperatura.

En capítulos posteriores se mostrará el software para la implementación e interpretación de todos estos comandos.

Sin embargo, aún existe un problema antes de poder emplear el BLE con Arduino. El chip BLE utilizado trabaja a 3.3V (tensión que puede ser proporcionada por el Arduino) y, en consecuencia, sus respuestas son también proporcionadas a 3.3V. Esto hace que Arduino no realice una correcta lectura de los datos recibidos desde el BLE, ya que funciona a 5V. Por tanto, se hace necesario un *voltaje shifter* (cambiador de tensión) que modifique las tensiones entre 3.3V y 5V de modo que la comunicación entre el BLE y el Arduino se produzca correctamente.

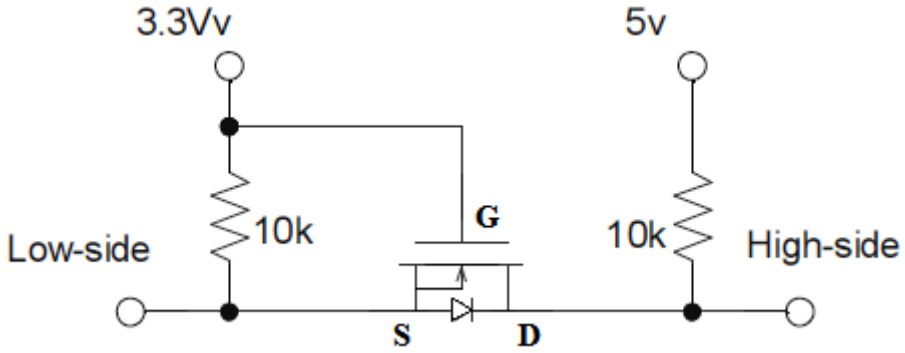


FIG. 36: VOLTAGE SHIFTER PARA EL BLE

En el circuito que se muestra en la imagen superior, cuando el "low-side" (BLE) transmite un 1 (3.3V) el mosfet se apaga y el "high-side" queda conectado a 5V a través de la resistencia. Sin embargo, cuando el "low-side" transmite un 0 (0V), el mosfet se activa y el "high-side" se baja a 0v.

De este modo, las respuestas enviadas desde el BLE hacia el Arduino cambian de 3.3V a 5V; realizándose así la correcta transmisión y recepción de datos.

Por tanto, el conexionado final entre Arduino, el *votage shifter* y el BLE queda del siguiente modo:

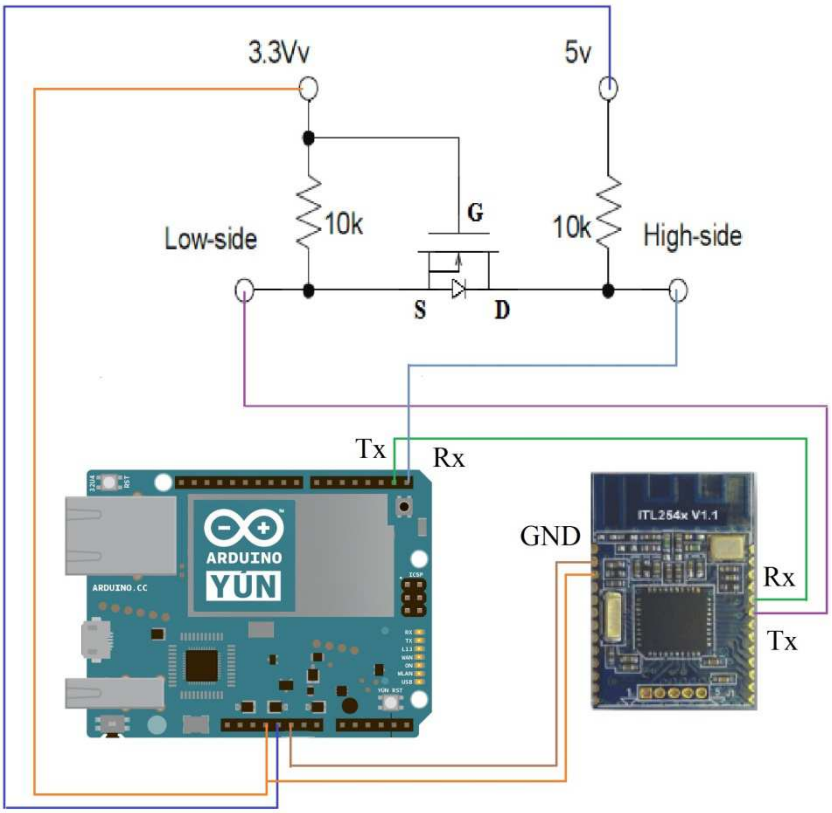


FIG. 37: ESQUEMA DEL CIRCUITO PARA SENSADO DE TEMPERATURA MEDIANTE BLE

Tras comprobar que el sistema funciona, se diseña la placa PCB para este módulo:

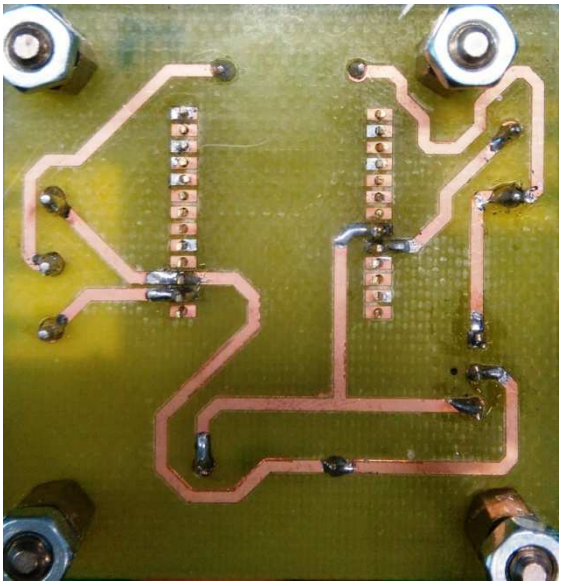


FIG. 38: PARTE IMPRESA DE LA PCB DEL MÓDULO PARA EL SENSADO MEDIANTE BLE

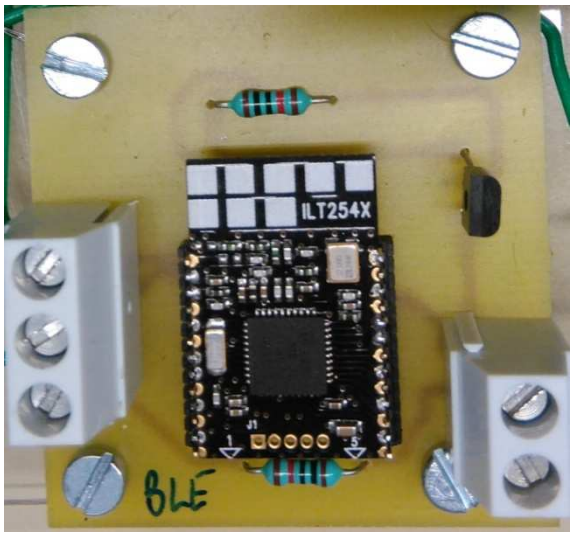


FIG. 39: VISTA SUPERIOR DE LA PCB DEL MÓDULO PARA EL SENSADO MEDIANTE BLE

3.1.4 CALIBRACIÓN Y COMPARATIVA DE LOS TRES MÓDULOS DE SENSADO

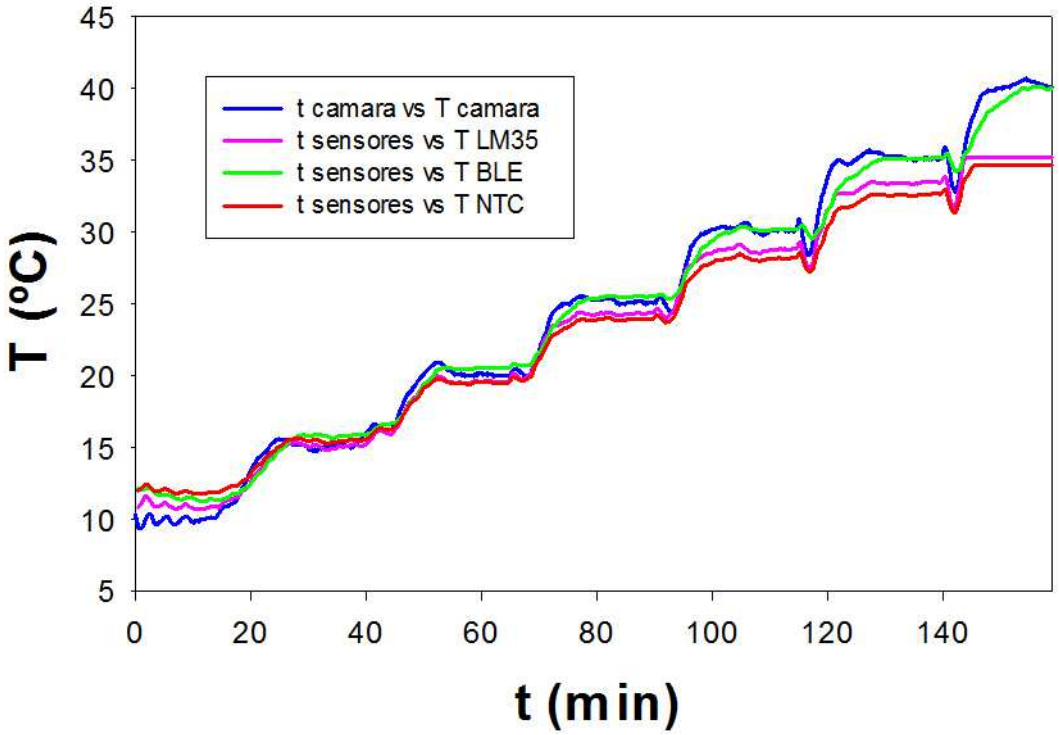
Tras el diseño e implementación de los tres módulos de sensado, se sitúan todos ellos en el interior de una cámara de temperatura controlada para comprobar el funcionamiento de cada uno de ellos y realizar las calibraciones necesarias.

Se desea comprobar la exactitud de los sensores tanto ante temperaturas constantes, como ante subidas o bajadas de temperatura. Además, puesto que el sistema completo está pensado para funcionar en temperaturas entre 10ºC y 40ºC, se diseña el ensayo en la cámara de temperatura controlada de modo que la temperatura dentro de ésta varíe del siguiente modo:

- Inicialmente: 10ºC
- 10ºC durante 15min
- Cambio de 10ºC a 15ºC a lo largo de 10min
- 15ºC durante 15min

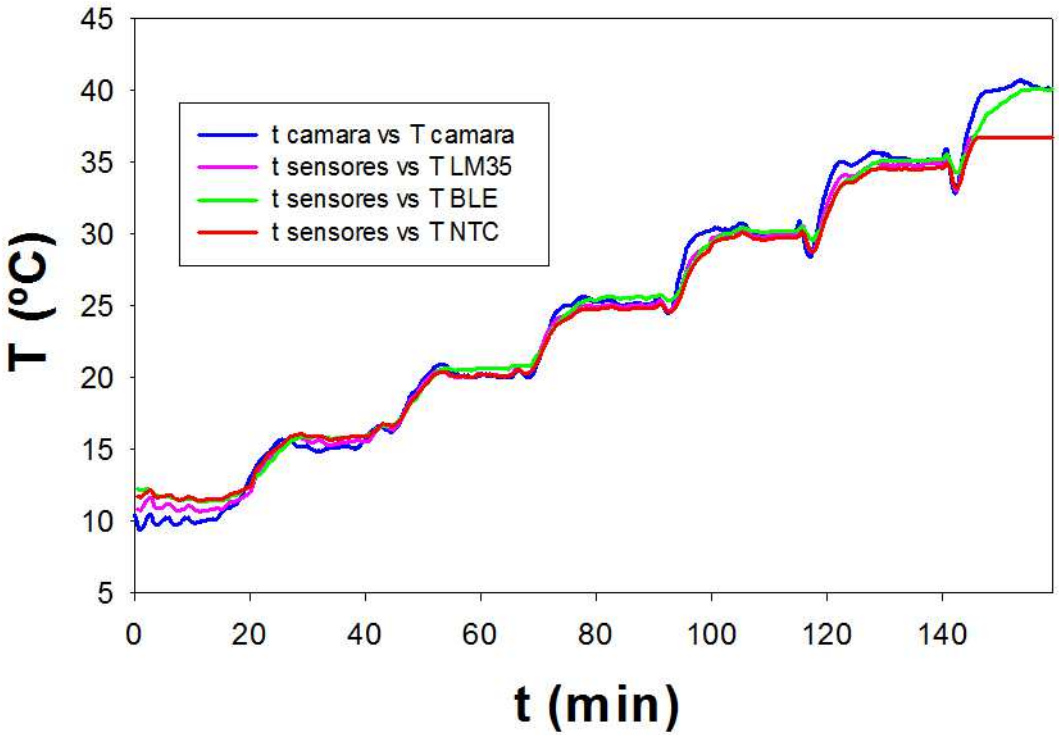
- Cambio de 15ºC a 20ºC a lo largo de 10min
- 20ºC durante 15min
- Cambio de 20ºC a 25ºC a lo largo de 10min
- 25ºC durante 15min
- Cambio de 25ºC a 30ºC a lo largo de 10min
- 30ºC durante 15min
- Cambio de 30ºC a 35ºC a lo largo de 10min
- 35ºC durante 15min
- Cambio de 35ºC a 40ºC a lo largo de 10min
- 40ºC durante 15min

La cámara empleada consta además de un sensor propio que permite saber la temperatura "exacta" en el interior de ésta en cada momento. Por lo que una vez terminado el ensayo es posible comparar los datos obtenidos por los distintos módulos con los datos reales de la cámara. En la siguiente imagen se muestran los resultados obtenidos inicialmente.



GRÁFICA 3: GRÁFICA T-T ANTERIOR A LA CALIBRACIÓN DE LOS SENSORES

Tras observar estos resultados y después de una pequeña modificación del software, los nuevos resultados obtenidos son los que se muestran en la siguiente imagen.



GRÁFICA 4: COMPARATIVA RESULTADOS SENSORES TRAS LA CALIBRACIÓN

En la imagen se pueden observar, fundamentalmente, dos cosas:

- Entre los 0 y los 20 minutos la temperatura medida por los distintos módulos es algo superior a la real de 10ºC. Esto se debe a que, inicialmente, la cámara de temperatura controlada se encuentra en torno a los 20ºC y durante el proceso de enfriamiento de ésta (tiempo que no está recogido en la gráfica) los sensores de los tres módulos se enfrían a menor velocidad que la cámara. Si la cámara se hubiera dejado más rato a 10ºC (40min en lugar de 20min, por ejemplo) los sensores habrían sido capaces de estabilizarse aunque hubiera sido a una temperatura algo mayor que la real como se observa que ocurre a 15ºC.
- En el tramo final se observa que las medidas realizadas por los módulos de la NTC y el LM35 llegan a un límite (en torno a los 36ºC) el cual no sobrepasan. Esto se debe a que, aunque teóricamente Arduino es capaz de "medir" tensiones de entrada de hasta 5V en sus pines analógicos; sin embargo, en este caso, no ocurre así. Como se explica en la nota de los anexos 1 y 2, la tensión real de alimentación del Arduino no es de 5V sino de unos 4.45V; por lo que Arduino es incapaz de "leer" tensiones de mayor valor en sus pines analógicos. Por tanto, cuando la temperatura hace que la tensión proporcionada por cada módulo supere los 4.45V, Arduino sigue midiendo 4.45V.

Además, se observa que las mediciones realizadas son bastante fieles a la realidad. A temperaturas bajas, las mediciones son algo superiores a la realidad; y a temperaturas elevadas, las mediciones resultan algo inferiores. Estos errores son cercanos a los 0.5ºC, lo cual, para el control de temperatura de una vivienda corriente, no resulta excesivo.

El BLE, el cual no necesita de ningún tipo de calibración extra en software ya que es el propio Beacon el que realiza la calibración y proporciona el dato final, tiene un comportamiento muy similar al de los módulos diseñados para el LM35 y la NTC.

3.2 SISTEMAS DE ACTUACIÓN

Los sistemas a emplear son: un relé, un optoacoplador, un motor paso a paso y un sistema de control analógico.

Los módulos de optoacoplador y control analógico se diseñan específicamente para este prototipo. Sin embargo, tanto el módulo de relé como el de motor PAP se adquieren listos para su utilización.

3.2.1 RELÉ

El módulo empleado es el SRD-05VDC-SL-C, el cual se puede observar en la siguiente imagen.

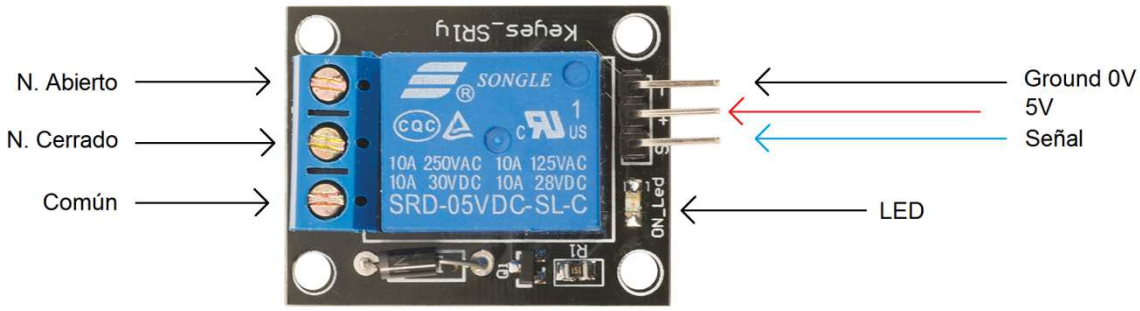


FIG. 40: MÓDULO RELÉ

Este módulo se alimenta a 5V (flecha roja), por lo que es perfecto para su utilización con Arduino. Además de los pines de tierra (GND) y alimentación (5V), dispone de un pin de señal, que permite el control del relé. El pin de señal se conecta a uno de los pines digitales de Arduino y, cuando se desea encender la caldera, se activa el relé enviando un "1" a través de dicho pin. Cuando se desea que la caldera esté apagada, se pone a "0" el pin de señal. El led que se puede ver en la imagen superior se enciende cuando la señal es "1" y se apaga cuando ésta es "0".

Existen tres pines de salida: común, NA (Normalmente Abierto: el contacto se encuentra abierto mientras la señal sea "0" y cerrado cuando la señal es "1") y NC(Normalmente Cerrado: el contacto se encuentra cerrado mientras la señal sea"0" y abierto cuando ésta es "1"). Para que la caldera se encienda, es necesario que ésta quede conectada a la red (220V AC), por lo que el pin que se ha de usar es el NA. De este modo, cuando la señal sea "0", el relé estará abierto y por tanto la caldera no estará

conectada a la corriente; y cuando la señal sea “1”, el relé se cerrará, cerrando el circuito entre la caldera y la red y, por tanto, activando la calefacción.

3.2.2 OPTOACOPLADOR

El optoacoplador empleado es el 4N25 cuyo esquema es el siguiente.

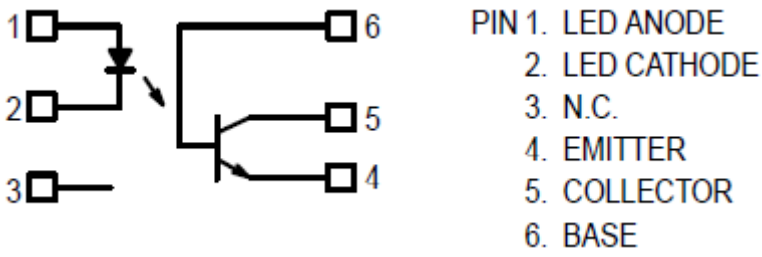


FIG. 41: ESQUEMA OPTOACOPLADOR 4N25

El pin 1 se conecta a uno de los pines digitales del Arduino y el 2 a GND, de modo que cuando el pin de señal de Arduino está en estado HIGH (“1”), pasa corriente por el diodo y éste emite luz activando el fototransistor. Sin embargo, el pin 1 no puede conectarse directamente a 5V ya que hay que limitar la corriente que atraviesa el diodo.

Según la hoja de características del optoacoplador, la corriente máxima a través del diodo entre los pines 1 y 2 es de 60mA, y la corriente característica (I_F) es de 10mA. Dado que la corriente máxima que puede proporcionar cada pin de Arduino es de 40mA, se decide diseñar el circuito de modo que la corriente a través del diodo sea de unos 10mA o menos. Para ello, se añade una resistencia entre la fuente (el pin de Arduino a 5V) y el ánodo del diodo.

En la página 2 de la hoja de características del optoacoplador se observa que la tensión típica en el ánodo del diodo es de 1.15V para I_F = 10mA. Por tanto, dado que la tensión proporcionada por Arduino es de 5V, entonces

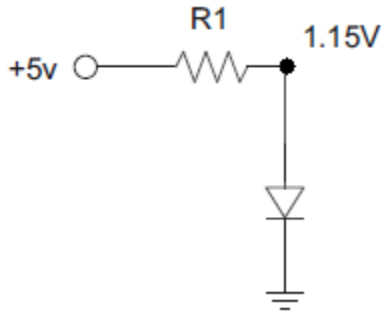


FIG. 42: CIRCUITO PARA EL CÁLCULO DE LA RESISTENCIA EN SERIE CON EL DIODO DEL OPTOACOPLADOR

Así, puesto que se desea una corriente de aproximadamente 10mA:

$$R_1 = \frac{5 - 1.15}{10 \cdot 10^{-3}} = 385\Omega$$

Sin embargo, ese no es un valor de resistencia normalizada y, al igual que en el resto de módulos, se desea emplear resistencias normalizadas; por lo que se decide que

$R_1 = 390\Omega$

El circuito final para el módulo optoacoplador es el siguiente:

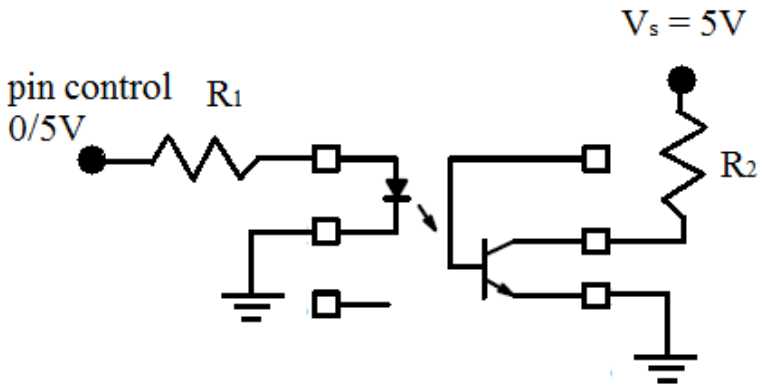


FIG. 43: CIRCUITO FINAL PARA EL CONTROL MEDIANTE OPTOACOPLADOR

La resistencia R_2 sirve para limitar la corriente por el colector del fototransistor. No se necesita una corriente elevada ya que, tal y como se explica en el apartado 2.2.1.2, el colector y el emisor del fototransistor se conectan a una entrada libre de potencial de la caldera, la cual no demanda corriente. Por tanto, a fin de obtener una corriente pequeña que pueda ser proporcionada sin problemas por Arduino, se selecciona una resistencia

$R_2 = 1k\Omega$

De modo que la corriente en R_2 es de unos 4.4mA.

Así, la PCB para el módulo optoacoplador resulta del siguiente modo:

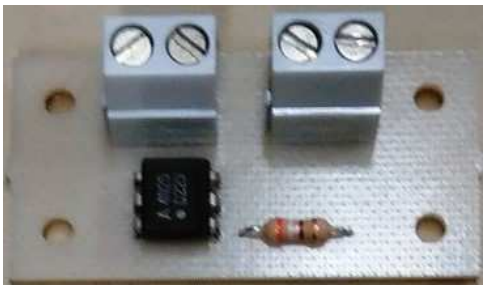


FIG. 44: VISTA SUPERIOR DE LA PCB PARA EL CONTROL MEDIANTE OPTOACOPLADOR

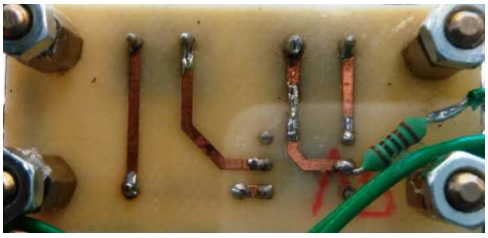


FIG. 45: VISTA INFERIOR DE LA PCB PARA EL CONTROL MEDIANTE OPTOACOPLADOR

3.2.3 Motor PAP (Paso a Paso)

El motor PAP empleado es el 28BYJ-48 con controlador ULN2003A

Fig. 46: Stepper motor 28BYJ-48 con controlador ULN2003A

Para este prototipo, se supone la utilización de una válvula de bola

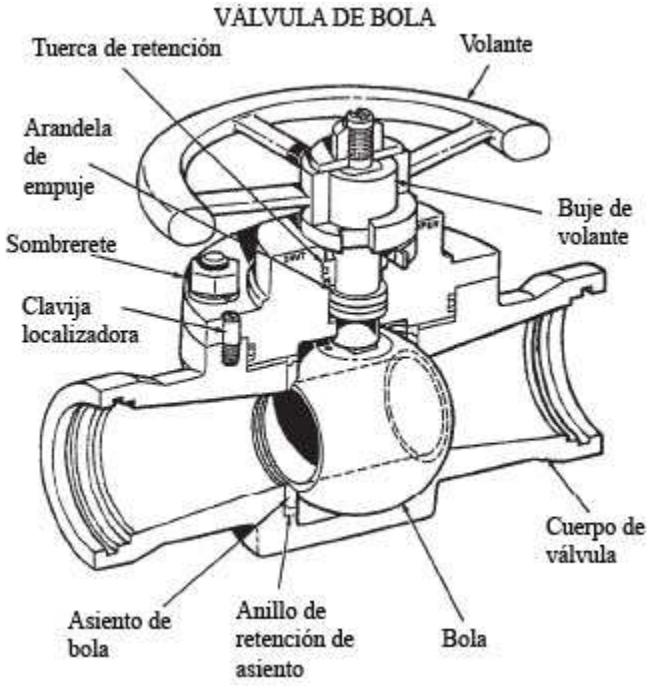


Fig. 47: Válvula de bola

La cual, si se encuentra en la posición de la imagen, deja pasar el agua, encendiendo la caldera. Sin embargo, si la bola está girada 90º, impide el paso del agua.

La relación de transmisión del motor 28BYJ-48 es

Gear Ratio = 64

Y el ángulo de paso

Stride Angle = 5.625º

Por tanto, el número total de pasos (en 360º) de este motor es

$$Pasos = n^{o}\, DePasosEn.1Vuelta \cdot GearRatio = \frac{360^{o}}{5.625^{o}} \cdot 64 = 4096$$

Así, si se desea que la caldera se encienda, es necesario abrir la válvula (girar el motor 90º) y, si se desea apagar la caldera, se debe cerrar la válvula (girar el motor 90º en sentido contrario).

Para girar el motor 90º, es necesario dar

$$\frac{4096\, pasos}{360^{o}} = \frac{pasos}{90^{o}} = 1024\, pasos$$

La secuencia de polarización de las bobinas que se ha de seguir para girar el motor es la que se muestra en la siguiente tabla.

Lead Wire Color	---> CW Direction (1-2 Phase)							
	1	2	3	4	5	6	7	8
4 ORG	-	-						-
3 YEL		-	-	-				
2 PIK				-	-	-		
1 BLU						-	-	-

FIG. 48: SECUENCIA DE POLARIZACIÓN DE LAS BOBINAS DEL MOTOR PAP

Cada cambio de polarización supone un avance de un paso. Por tanto, dado que la secuencia consta de 8 cambios (8 avances o pasos) y que se desea avanzar 1024 pasos (90º), entonces es necesario repetir la secuencia

$$\frac{1024}{8} = 128 \text{ veces cada vez que se desea abrir o cerrar la válvula}$$

Por otra parte, si se desea abrir la válvula, la secuencia se debe realizar en el orden que aparece en la tabla (1-8) para girar la bola 90º en sentido horario; mientras que si se desea cerrarla, la secuencia se ha de seguir en orden inverso (8-1), para girar la bola 90º en sentido antihorario. En el apartado de software se mostrará cómo se realiza este control.

En el caso de que se empleara un tipo distinto de válvula bastaría con modificar el software de modo que se giren el número de grados necesarios en cada caso.

3.2.4 CONTROL ANALÓGICO

Como ya se ha explicado en el apartado 2.2.2, este tipo de control consiste en modificar la temperatura del agua de calefacción mediante el control de la tensión de alimentación de la caldera. Esta tensión de alimentación suele variar entre 0 y 10V.

Arduino no dispone de pines analógicos de salida que permitan proporcionar una tensión determinada. Sólo posee pines digitales que proporcionan, o bien "0s" (0V) o bien "1s" (5V). Por eso, se hace necesario buscar otro método para proporcionar dichas tensiones.

Arduino puede proporcionar señales PWM (Pulse Width Modulation). Una señal PWM es una onda cuadrada en la que se puede modificar el tiempo durante el cual ésta se encuentra en estado "HIGH" (Duty Cycle).

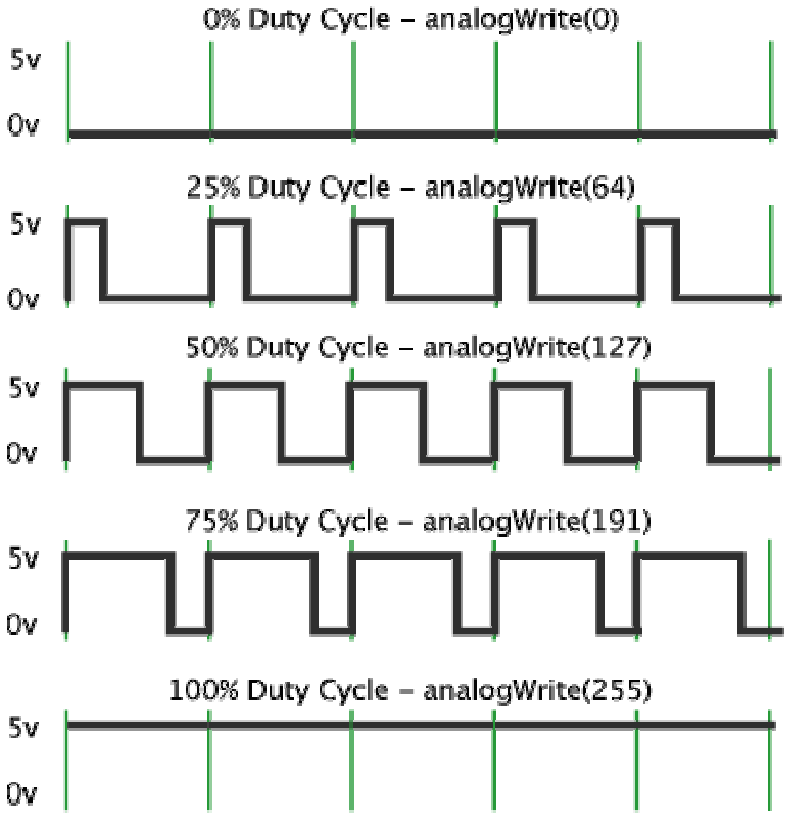


FIG. 49: EJEMPLO DE SEÑAL PWM

El tiempo "on" de la señal con respecto al periodo total de ésta es el Duty Cycle, el cual se puede expresar de la siguiente manera:

$$D = \frac{w}{T} \qquad \textbf{[17]}$$

Donde:

w: tiempo que la señal está en estado "HIGH"

T: periodo de la señal

En la Fig. 49 se muestran 5 señales PWM distintas.

- La primera señal tiene un Duty Cycle del 0%; es decir, la señal se encuentra en estado LOW todo el tiempo. De este modo, el valor medio de la señal es de 0V.

- La segunda señal tiene un Duty Cycle del 25%; por lo que la señal se encuentra en estado HIGH durante ¼ del periodo. Así, el valor medio de la señal es de 1.25V
- En la tercera señal D = 50%. Por tanto, la señal está a 5V durante la mitad del periodo, siendo el valor medio de la señal de 2.5V.
- La cuarta señal tiene un Duty Cycle del 75%, generando una señal cuyo valor medio es de 3.75V
- La última señal tiene un D = 100%. Por tanto, la señal se encuentra en estado HIGH durante todo el periodo, creando una señal de valor medio 5V.

Por tanto, se puede emplear este tipo de señal para generar las señales de entre 0 y 10V necesarias para el control de las calderas.

Para poder realizar el control analógico mediante señal PWM es necesario seguir los siguientes pasos:

1) Primero, se ha de calcular el valor de tensión necesario en cada momento para alimentar la caldera a fin de alcanzar la temperatura deseada. Para ello, se implementa (en software) un PI (control proporcional integral) que permite determinar el valor de la tensión necesaria en cada momento en función de la diferencia entre la temperatura medida y la temperatura de consigna (error).
2) La caldera debe ser alimentada con una señal continua. Sin embargo, las señales PWM generadas por Arduino son señales cuadradas. Por tanto, es necesario un filtro que permita obtener una señal continua cuyo valor sea el valor medio de la señal PWM.
3) Las señales PWM de Arduino varían entre 0 y 5V, ya que Arduino no puede proporcionar más de 5V. Por tanto, el máximo valor medio de sus señales PWM es de 5V (cuando el Duty Cycle sea igual al 100%). No obstante, la caldera debe ser alimentada entre 0 y 10V. En consecuencia, se hace necesario un circuito de amplificación que amplifique la señal continua de salida del filtro.

3.2.4.1 Controlador PI

Como ya se ha explicado, este controlador permite conocer el valor de tensión de alimentación de la caldera necesario en cada momento a fin de conseguir la temperatura deseada.

El sistema de control a emplear es un sistema en lazo cerrado como el que se muestra en la siguiente figura.

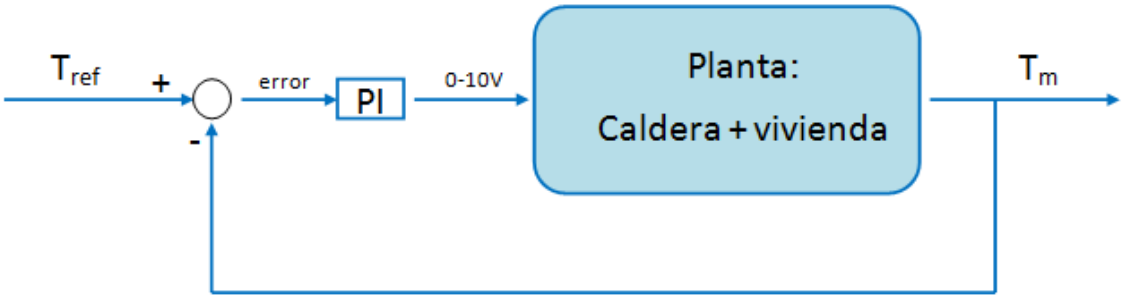


FIG. 50: LAZO DE CONTROL PARA EL CONTROL ANALÓGICO

La forma general de un controlador PI es la siguiente:

$$u(V) = K_p \cdot e(t) + \frac{K_p}{T_i}\int e(t)dt = K_p \cdot e(t) + I(t) \qquad \textbf{[18]}$$

Donde:

u(V): acción de control

K_p: constante proporcional. Crea una acción de control proporcional al error.

T_i: Constante integral. Proporcional al tiempo en el que se desea que se elimine el error en estado estacionario.

Para diseñar el controlador PI son necesarios dos pasos:

- Discretizar el PI: modificar la ecuación del PI para sustituir las variables continuas (empleadas en controladores analógicos) por variables muestreadas que se puedan emplear en un controlador digital (software)
- Sintonizar el PI: determinar los valores de las constantes K_p y T_i características de cada controlador. Estas variables no dependen sólo del controlador sino también de la planta sobre la que éste actuará. Es decir, K_p y T_i tendrán valores distintos para distintas viviendas.

❖ Discretización del PI

La ecuación [18] consta de dos partes: una correspondiente a la parte proporcional del controlador ($K_p \cdot e(t)$), y otra correspondiente a la parte integral (I(t)). La discretización se puede realizar también en dos partes:

- *Acción proporcional (P)*

$$K_p \cdot e(t) \Rightarrow K_p \cdot e(t_k) \qquad [19]$$

- *Acción integral (I)*

Existen varios métodos para la discretización de la acción integral: diferencia hacia atrás, diferencia hacia delante, aproximación de Tustin…

La aproximación de Tustin es una de las más precisas, por lo que es la empleada en este caso.

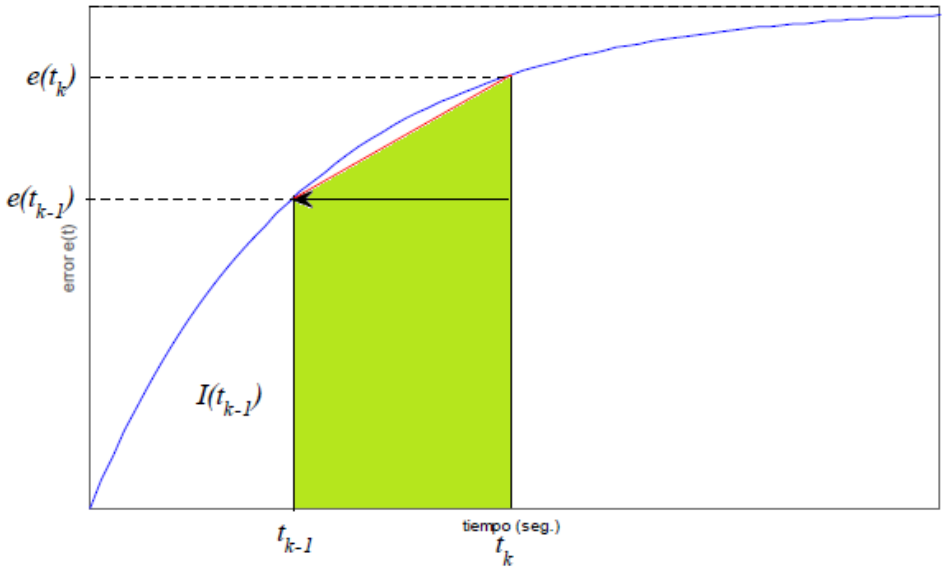


FIG. 51: APROXIMACIÓN DE TUSTIN PARA LA INTEGRAL DEL PI

La integral del error entre los instantes "t_{k-1}" y "t_k" es igual al área debajo de la curva. Para aproximar esa área, se emplea el área de la zona sombreada en verde. Por tanto

$$I(t_k) = I(t_{k-1}) + \frac{K_p \cdot h}{T_i} \cdot \frac{e(t_k) + e(t_{k-1})}{2} \qquad \textbf{[20]}$$

- Discretización final del PI

Tras discretizar por separado la parte proporcional y la integral, el controlador completo queda como sigue:

$$u_k(V) = K_p \cdot e(t_k) + I(t_k) = K_p \cdot e(t_k) + I(t_{k-1}) + \frac{K_p \cdot h}{T_i} \cdot \frac{e(t_k) + e(t_{k-1})}{2} \qquad \textbf{[21]}$$

❖ Sintonización del PI

Tras hallar la forma discreta del controlador PI a utilizar, es necesario calcular los valores de las constantes K_p y T_i. Estos valores son calculados mediante el método de Ziegler Nichols (ZN).

Para poder sintonizar el PI es necesario registrar la respuesta de la planta (caldera+vivienda) ante una entrada escalón unitario (1V de tensión de alimentación de la caldera) en lazo abierto. Para obtener una curva más representativa, se puede realizar el ensayo con una entrada de 10 voltios (caldera al máximo de su funcionamiento), y después dividir el valor de "a" (Fig. 53) por 10.

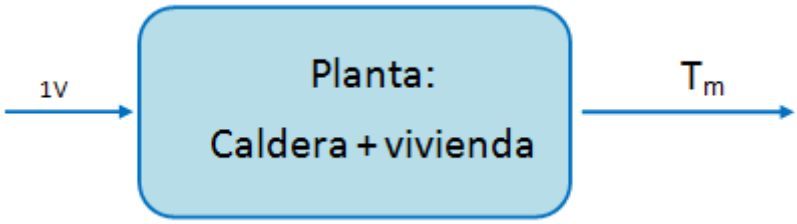


FIG. 52: LAZO ABIERTO DE CONTROL PARA LA SINTONIZACIÓN DEL PI

La respuesta del sistema suele ser similar a la que se muestra en la figura siguiente.

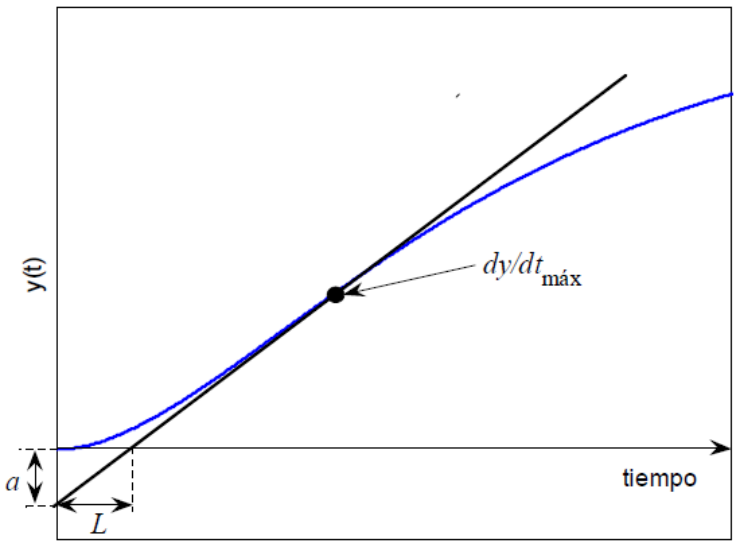


FIG. 53: MÉTODO DE SINTONÍA ORIGINAL DE ZN (ZIEGLER NICHOLS) PARA LA SINTONIZACIÓN DEL PI POR ENTRADA ESCALÓN UNITARIO

Una vez obtenida la curva de la respuesta (curva azul), se traza una recta tangente a ésta por su punto de máxima pendiente y se obtienen los valores "a" y "L" que se observan en la figura superior. Tras hallar estos valores, según el método de ZN, las constantes del PI son los siguientes:

$$K_p = 0.9/a \qquad [22]$$

$$T_i = 3 \cdot L \qquad [23]$$

Por tanto, realizando el ensayo mostrado en las Fig. 52 y Fig. 53 se obtendrían fácilmente dichas constantes. Sin embargo, durante la realización de este proyecto, no se dispone de ninguna vivienda con este tipo de caldera en la cual se pueda realizar el ensayo. Además, este análisis debe ser realizado individualmente en cada vivienda en la que se implemente el sistema, ya que los resultados obtenidos varían de unos hogares a otros dado que la temperatura no varía igual, por ejemplo, en habitaciones pequeñas que en habitaciones grandes.

En este proyecto, a la hora de implementar el software correspondiente a este módulo, se tomarán valores estimados de K_p y T_i basados en varios artículos científicos.

Pero debe quedar claro que en el momento de instalar este sistema en cualquier vivienda se debe realizar el ensayo anteriormente descrito a fin de obtener los valores adecuados de las constantes del PI para ese caso concreto.

Una vez implementado en software el controlador y conocida la tensión necesaria, se determina el Duty Cycle necesario para obtener con Arduino una señal PWM con valor medio igual al valor obtenido mediante el PI y se programa el software necesario para la creación de esa señal.

En apartados posteriores se muestra y explica el software empleado para la implementación del controlador PI así como para la creación de las correspondientes señales PWM.

3.2.4.2 Filtro para la señal PWM

Para convertir las señales cuadradas PWM en señales continuas cuyo valor sea igual al valor medio de las PWM, es necesario emplear un filtro RC.

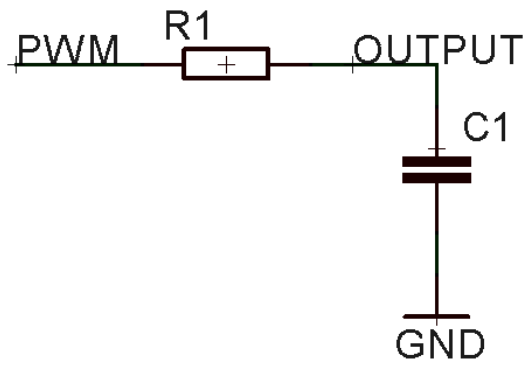


FIG. 54: FILTRO RC PARA LA SEÑAL PWM

Mientras la señal PWM está "HIGH" el condensador se carga, y cuando ésta es "LOW", el condensador se descarga a través de la resistencia R_1. De este modo, se obtiene una señal de salida como la de la siguiente imagen.

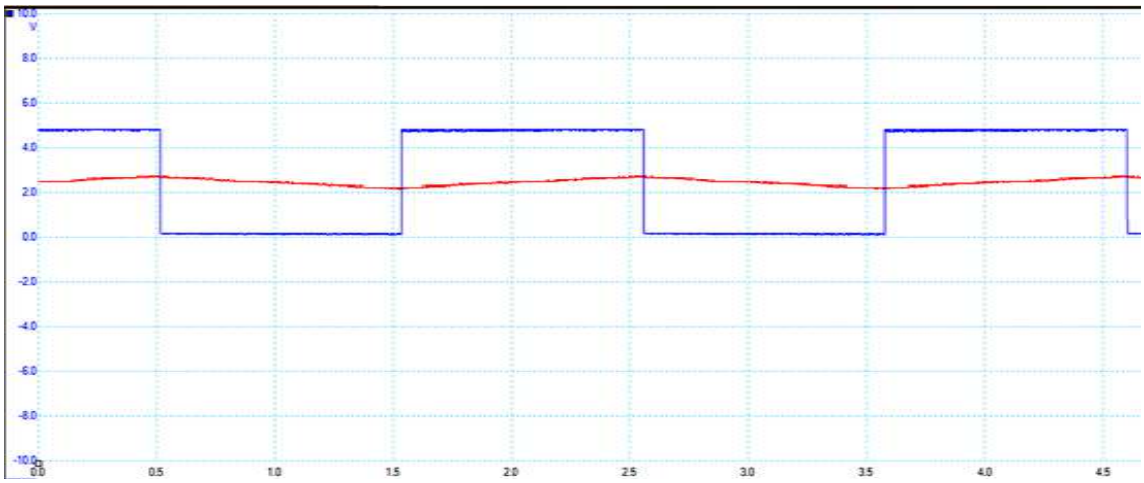


FIG. 55: SALIDA DE UN FILTRO RC APLICADO A UNA SEÑAL PWM

Dependiendo de los valores de R_1 y C_1 empleados en el filtro de la Fig. 54 se obtiene un valor de rizado diferente en la señal de salida del filtro (roja). El filtro debe tener una frecuencia de corte lo suficientemente baja como para eliminar las componentes no deseadas. Generalmente, se procura que la frecuencia de corte del filtro sea, por lo menos, 10 veces inferior a la frecuencia de la señal filtrada.

Las señales PWM generadas por Arduino tienen una frecuencia:

$$f_{Arduino} = 490Hz$$

Por tanto, dado que en un filtro generalmente se busca una atenuación mínima de una década, lo ideal sería obtener una frecuencia de corte:

$$f_c \leq \frac{f_{señal.filtrada}}{10} = \frac{f_{Arduino}}{10} = \frac{490}{10} = 49Hz \qquad [24]$$

Tras comprobar el comportamiento del filtro para distintos valores de R_1 y C_1, se decide emplear

$R_1 = 150\Omega$

$C_1 = 100\mu F$

Ya que estos valores cumplen

$$f_c = \frac{1}{2\pi RC} = \frac{1}{2\pi \cdot 150 \cdot 100 \cdot 10^{-6}} = 11Hz \leq 49Hz \qquad [25]$$

Y proporcionan una señal de salida con un factor de rizado bajo, tal y como se observa en la siguiente imagen.

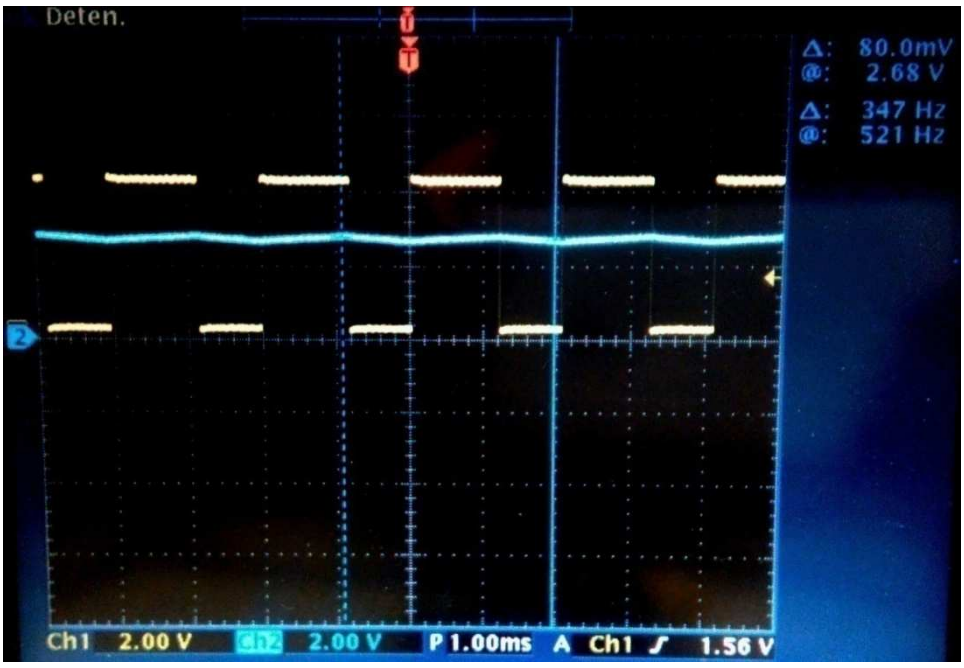


FIG. 56: SEÑAL PWM (AMARILLA) Y SEÑAL A LA SALIDA DEL FILTRO (AZUL)

3.2.4.3 Amplificación de la señal de salida del filtro

Como ya es sabido, la tensión máxima que puede proporcionar Arduino es de 5V. Por tanto, el máximo valor medio de la señal PWM generada por éste será de 5V (cuando el duty cycle sea del 100%). Sin embargo, tal y como se comentaba al inicio de la sección 3.2.4, la tensión necesaria en este módulo para alimentar la caldera debe variar entre 0 y 10V. En consecuencia, es necesario amplificar la tensión de salida del filtro RC para convertirla a los valores de tensión necesarios.

Las tensiones máxima y mínima obtenidas a la salida del filtro RC son:

$$V_{RC.máx} = 5V$$

$$V_{RC.mín} = 0V$$

Para obtener señales de entre 0 y 10V es necesaria una ganancia G = 2. Aunque hasta el momento en este proyecto se han empleado amplificadores de instrumentación, en este caso no es posible emplear los INA126PA de los que se dispone; ya que su ganancia mínima es G = 5. Por tanto, se emplea un amplificador operacional (UA741CN) en configuración de circuito amplificador no inversor.

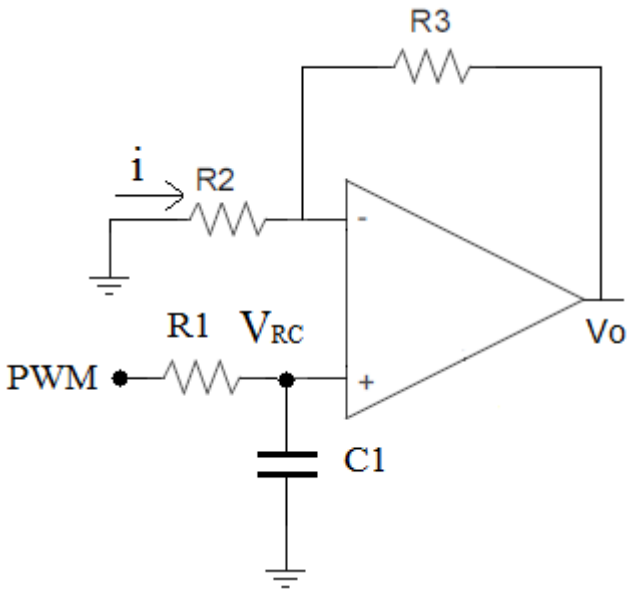


FIG. 57: FILTRADO Y AMPLIFICACIÓN DE LA SEÑAL PWM PARA EL CONTROL ANALÓGICO

Donde:

$$V_0 = V_{RC} \cdot \left(1 + \frac{R_3}{R_2}\right) \Leftrightarrow G = \frac{V_0}{V_{RC}} = 1 + \frac{R_3}{R_2} \qquad [\,26\,]$$

Por tanto, dado que se necesita una G = 2, entonces

$$2 = 1 + \frac{R_3}{R_2} \Leftrightarrow R_2 = R_3 \qquad \textbf{[27]}$$

Además, se desea que la corriente "i" sea pequeña. Así, se calculan los valores de R_2 y R_3 de modo que $i_{máx} \approx 1mA$:

$$i_{máx} = \frac{V_{RCmáx}}{R_2} \Rightarrow 1 \cdot 10^{-3} = \frac{5}{R_2} \Leftrightarrow R_2 = 5k\Omega \qquad \textbf{[28]}$$

A fin de emplear valores normalizados de resistencias y para cumplir la ecuación [26], entonces:

$R_2 = R_3 = 4k7\Omega$

Por último, al circuito de la Fig. 57 es necesario añadir un conversor DC-DC para poder alimentar el amplificador operacional empleado. Este AO requiere una alimentación bipolar de entre 10 y 15V, y para esto se emplea el conversor MCA05D12S, que proporciona, a partir de los 5V de Arduino, una tensión de ±12V.

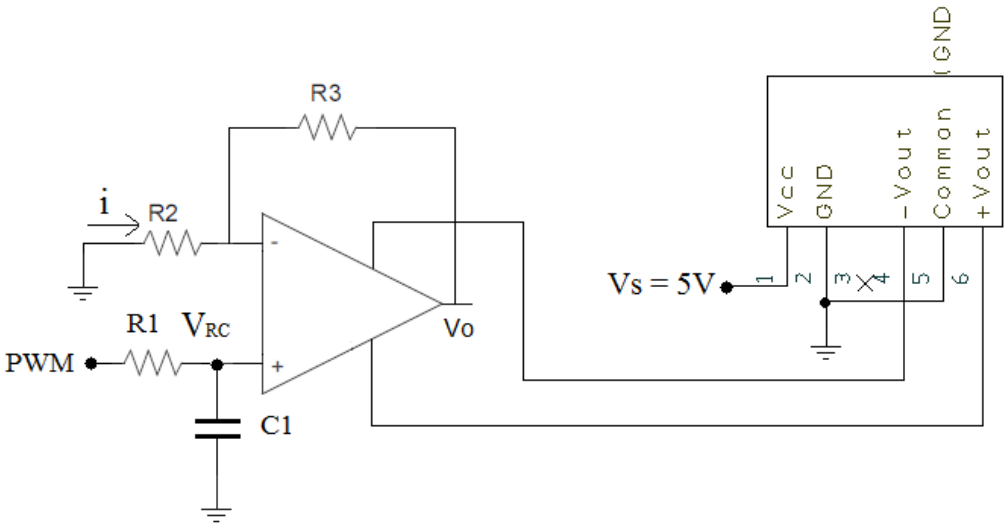


FIG. 58: CIRCUITO ANALÓGICO FINAL

Así, tras el filtrado y la amplificación de la señal PWM, se obtiene la siguiente señal, respecto a la misma PWM empleada en la Fig. 56.

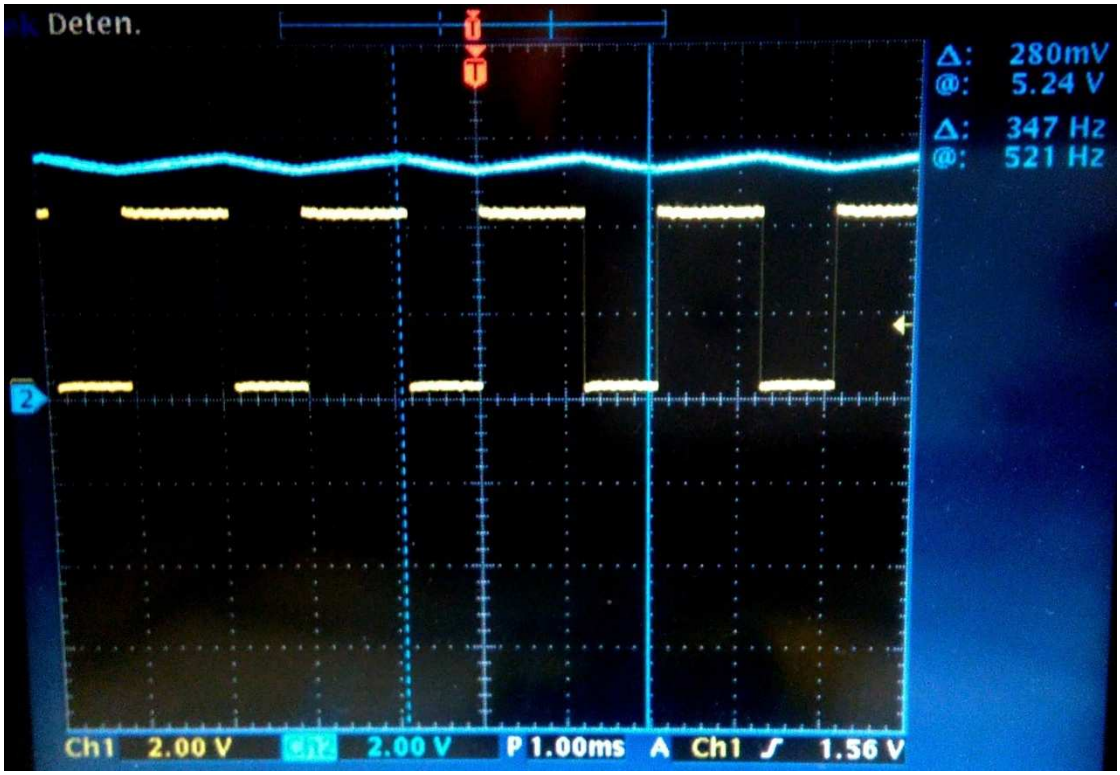


FIG. 59: FILTRADO Y AMPLIFICACIÓN DE LA SEÑAL PWM PARA EL CONTROL ANALÓGICO

Como se puede observar en la imagen superior, la señal final obtenida es aproximadamente una señal continua de valor igual al valor medio de la señal PWM. En la Fig. 56, el valor medio de la señal filtrada es de 2.68V y en la Fig. 59, el valor medio de dicha señal una vez amplificada es de 5.24V. Por tanto, la ganancia real obtenida es muy similar a la deseada G = 2.

A continuación se muestra la PCB creada para el presente módulo:

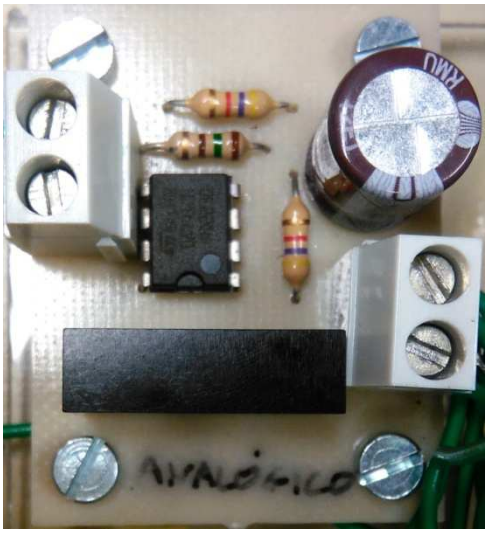


FIG. 60: VISTA SUPERIOR DE LA PCB PARA EL CONTROL ANALÓGICO

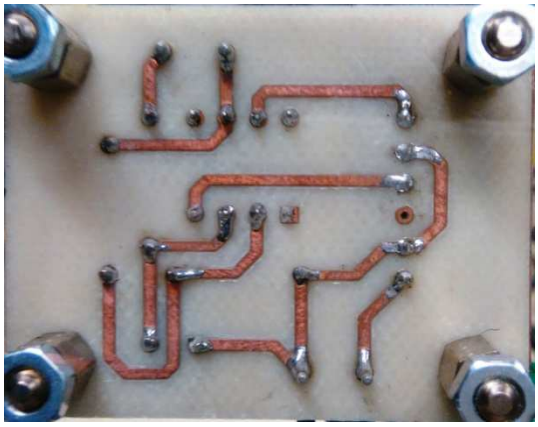


FIG. 61: VISTA INFERIOR DE LA PCB PARA EL CONTROL ANALÓGICO

3.3 COMPROBACIÓN DEL FUNCIONAMIENTO CONJUNTO DE LOS DISTINTOS MÓDULOS

Tras diseñar y desarrollar todos los módulos hasta el momento expuestos, y antes de incorporarlos en el prototipo final, se procede a la comprobación del correcto funcionamiento de todos los posibles pares sensor-actuador junto con el software para el control remoto. Es decir, se comprueba que todos los sensores son capaces de actuar con cada uno de los actuadores y, al mismo tiempo, permitir el control remoto del sistema completo.

Todos los módulos funcionan a la perfección en conjunto, tal y como lo hacían por separado durante el desarrollo de cada uno de ellos. Sin embargo, surge el siguiente problema:

El Arduino empleado hasta el momento es Arduino Yún, ya que es el único que incorpora conexiones Wi-Fi y Ethernet sin necesidad de añadirle ninguna *shield*. Esta placa da muy buenos resultados para todos los módulos diseñados; así como para el control remoto a través de un servidor web. Sin embargo, como ya se comentaba en el apartado 2.3, Arduino Yún contiene dos microprocesadores: uno para las funciones habituales del microprocesador en las placas Arduino y otro para controlar las conexiones Ethernet y Wi-fi, la tarjeta SD, etc. El problema surge al intentar controlar el sistema de manera remota empleando el BLE como módulo sensor.

Como se explica en el apartado 3.1.3, el BLE se comunica con Arduino a través de los pines Serial (Rx y Tx). Pues bien, el enlace de hardware entre los dos microcontroladores de Arduino Yún es exactamente el mismo para el control de los pines serie (Rx y Tx) que para el control de las conexiones Wifi y Ethernet. Por tanto, resulta imposible emplear al mismo tiempo los pines Serial y las conexiones Ethernet o Wi-Fi.

Para solucionar este problema, se decide desechar la idea inicial de emplear Arduino Yún y utilizar en su lugar Arduino Uno junto con una *shield Ethernet* que permita la conexión del primero a internet.

Así pues, el prototipo final consta de todos los módulos expuestos anteriormente y de Arduino Uno junto con una *Shield Ethernet*.

3.3 ENSAMBLAJE DE LOS DISTINTOS MÓDULOS EN EL PROTOTIPO FINAL

Una vez diseñados los 7 módulos:

- Sensado de temperatura:
  - NTC
  - LM35
  - BLE
- Actuación:
  - Relé
  - Optoacoplador
  - Motor PAP
  - Control analógico

y comprobado el correcto funcionamiento de todos ellos tanto en solitario como conjuntamente; se procede al ensamblaje de cada uno de estos módulos dentro del prototipo final de termostato. Los módulos de control analógico, relé, BLE y optoacoplador son incorporados dentro de lo que posteriormente será el termostato. Sin embargo, la NTC, el LM35 y el motor no son incluidos dentro de éste. La NTC y el LM35 se colocan fuera del termostato a fin de poder situarlos en lugares más estratégicos para la medición de temperatura, cableándolos a distintos conectores del termostato. El motor debe estar unido a la válvula la cual debe girar, por lo que también es necesario situarlo en el exterior del termostato y conectarlo, igual que los sensores, a sus correspondientes conectores en el interior del termostato. La placa de Arduino también queda en el interior del termostato.

Las conexiones necesarias entre todos los distintos módulos se podrían realizar mediante el diseño de una nueva PCB en la que ensamblar las demás o una PCB general que incluyera todas lás demás y las conexiones necesarias entre ellas y Arduino. Sin embargo, para soldar el Arduino a ésta PCB sería necesario desoldar sus conectores, lo cual inutilizaría Arduino para posibles futuros proyectos. Por tanto, a fin de poder emplear la placa Arduino Uno en el futuro, se decide realizar todas las conexiones mediante cables en lugar de soldaduras, de modo que posteriormente sea posible desconectar cada módulo (incluido Arduino) del prototipo y emplearlo para nuevos fines.

Se diseña finalmente el prototipo final situando cada módulo en la posición óptima para obtener un termostato lo más pequeño posible. Además, todos los módulos se sitúan de modo que los conectores necesarios para módulos en el exterior queden al

borde del termostato, pudiendo así acceder fácilmente a ellos. Una vez diseñado el prototipo, se atornillan todos los módulos a una base de metacrilato de acuerdo al diseño y se realizan todas las conexiones pertinentes para el correcto funcionamiento de todo el sistema.

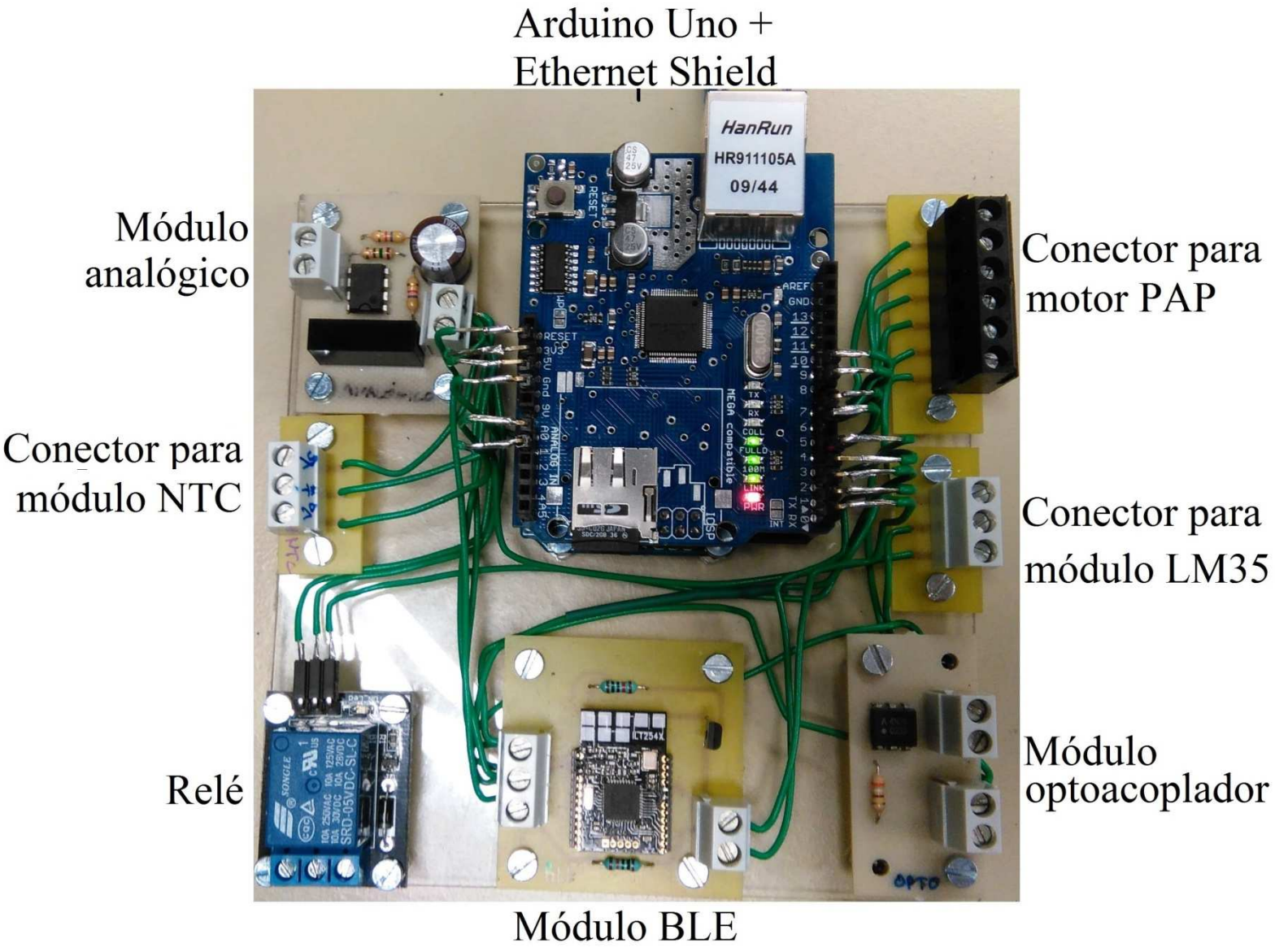


FIG. 62: PROTOTIPO FINAL

4. DESARROLLO DEL SOFTWARE

Al igual que el hardware, el software se desarrolla e implementa de manera modular. Así, se desarrollan distintas funciones para cada módulo, cada una de las cuales se ejecuta sólo si el correspondiente módulo está siendo empleado.

En el anexo 4 se muestra el software completo del termostato. A continuación se procede a explicar las distintas funciones de dicho software.

Lo primero que se observa en el *skecth* es un array de configuración. Este array ha de ser configurado en el momento de la instalación del termostato, de modo que el programa sepa que módulos se están empleando. A continuación se encuentra toda la parte de configuración, en la que se determinan variables del programa y se establecen los módulos a emplear en función del array de configuración.

Después de la configuración, aparece el *void loop*, la parte del programa que se repetirá continuamente mientras éste esté ejecutándose; es decir, mientras el termostato esté encendido.

Lo primero que aparece es el software para la creación de la página web alojada en el servidor de Arduino, a la cual el usuario accederá para conocer y modificar los datos y el estado de la caldera. Lo que hace esta parte del software es crear una página web en la que se muestra la temperatura de la vivienda, la temperatura de consigna y el estado de la calefacción (on/off) y se posibilita al usuario modificar la temperatura de consigna del hogar. La página web visualizada por el usuario es la que se muestra a continuación:

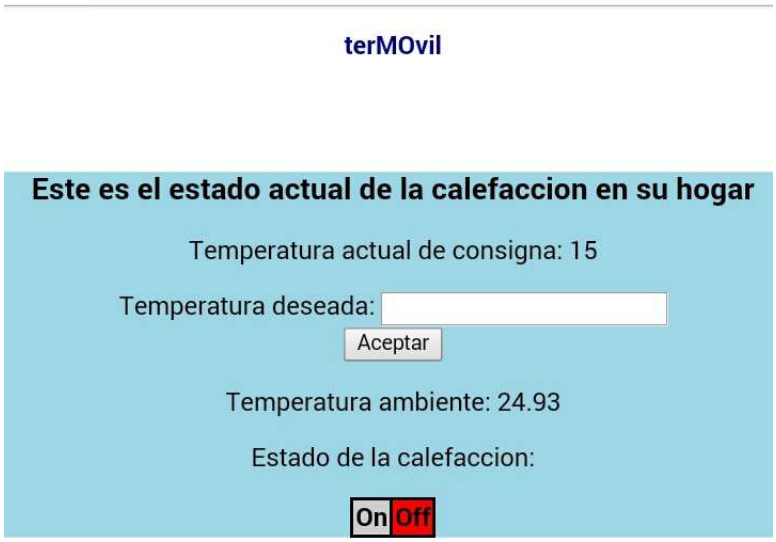


FIG. 63: PÁGINA WEB PARA EL CONTROL REMOTO DE SISTEMA

Después de esto, el programa determina el sensor y el actuador a emplear y llama a la función correspondiente. A continuación se explica el funcionamiento de cada una de estas funciones:

4.1 Función NTC

El pin de Arduino empleado para la lectura de la ”temperatura” (tensión) medida con la NTC es el pin A0. Para tener una medida más real, y dado que cada medición con NTC lleva tan solo unos ms, se realizan 100 medidas y se toma como medida de tensión la media de estas 100. Así, se obtiene el valor de tensión de salida del circuito del módulo NTC (V_0 en la ecuación [10]). Sin embargo, Arduino proporciona ese dato en bits; por lo que, para conocer el valor en voltios, es necesario multiplicar el valor inicial por la resolución de Arduino. Una vez conocido el valor de V_0 en voltios, para convertir este dato de tensión a temperatura, y dado que se conocen los valores de G y V_2, se aplican las fórmulas de las ecuaciones [10] y [5].

4.2 Función LM35

El pin de Arduino empleado para la lectura de la tensión de salida del módulo LM35 es el pin A1. Al igual que al realizar la medición con NTC, se realizan 100 medidas y se toma la media de todas ellas. El valor obtenido a través del pin A1 es también en bits, por lo que se hace necesario multiplicarlo por la resolución de Arduino para convertirlo a voltios. Una vez se tiene la tensión V_0 en voltios, se obtiene el valor de la temperatura mediante la ecuación [16].

4.3 Función BLE

Como ya se comentaba en la sección 3.1.3, para la adquisición de la temperatura mediante el BLE es necesario enviar desde Arduino una serie de comandos para obtener respuestas del Beacon.

Los comandos a enviar son los siguientes:

1) *trama_init* (*GAP_DeviceInit* en BTool): inicializa el BLE para poder comenzar la comunicación con Beacons.
2) *trama_discovery* (*GAP_DeviceDiscoveryRequest* en BTool): busca Beacons en las proximidades a los que poder conectarse y con los que interactuar.
3) *trama_establish* (*GAP_EstablishLinkRequest* en BTool): se conecta al Beacon encontrado con el comando *GAP_DeviceDiscoveryRequest* para poder comenzar con la comunicación.
4) *trama_temperatura* (*GATT_ReadUsingCharUUID*): solicita al Beacon el dato de temperatura mediante su UUID (Universally Unique Identifier).
5) *trama_terminate_Link* (*GAP_TerminateLinkRequest*): se desconecta del Beacon para que éste pueda estar accesible a otros BLEs. Se desconecta además para reducir el consumo; ya que los Beacons consumen muy poca energía mientras están esperando a que algún BLE se conecte a ellos.

Así pues, en la función BLE, se manda cada uno de estos comandos en el orden correspondiente y se comprueban las respuestas:

Tras enviar la *trama_init*, se comprueba que la respuesta obtenida es igual a *resp_trama_init* y si no es así se vuelve a enviar el comando

Como se puede ver en el anexo 3, el comando *trama_discovery* da lugar a 4 respuestas o eventos (0x067F (GAP_HCI_ExtentionCommandStatus), 0x060D (GAP_DeviceInformation), 0x060D (GAP_DeviceInformation) y 0x0601 (GAP_DeviceDiscoveryDone)). El comando que interesa en este caso es el último, ya que proporciona la dirección (Address) del Beacon o baliza encontrado y al cual nos querremos conectar. Por tanto, se leen las cuatro respuestas y cuando se llega a la última, se guarda la dirección del Beacon (datos 7 al 14) en la variable *resp_trama_discovery*. (En Arduino, los elementos de un array se empiezan a contar en 0; por lo que el dato 7 es en realidad el 8, el 8 es el 9, etc.)

Para conectarse al Beacon encontrado es necesario enviar la *trama_establish* para ese Beacon en particular. Por ello, se envía el comando *GAP_EstablishLinkRequest* que se puede observar en el anexo 3; pero cambiando los 6 últimos datos por la dirección obtenida mediante el comando *trama_discovery.*

Para comprobar que la conexión entre el Beacon y el BLE ha sido realiza con éxito, es necesario comprobar la segunda de las respuestas producidas por este comando. La respuesta obtenida ha de ser igual a la que se observa en el anexo 3, pero los datos 7 al 12 han de ser los correspondientes a la dirección del Beacon al que se pretende conectar. Por eso, antes de leer las respuestas al comando *trama_establish* se modifica el array *resp_trama_establish* para que concuerde con la dirección del Beacon deseado; y después se comprueba que la respuesta obtenida es igual a *resp_trama_establish*. Si la respuesta y *resp_trama_establish* concuerdan, entonces la conexión se ha establecido correctamente.

Una vez establecida la conexión, se debe solicitar el dato de temperatura. Esto se hace mediante el comando *trama_temperatura*, en la que los dos últimos datos (B7 FF) corresponden a la UUID del dato de temperatura.

En ocasiones, debido a problemas propios del Beacon, las respuestas a este comando no son correctas; por lo que, lo primero que se debe hacer al recibir la respuesta al comando *trama_temperatura*, es comprobar que la respuesta es correcta. La respuesta es correcta (no hay errores) si el dato correspondiente a status es 0x00 (esto se puede observar en el anexo 3, en la primera respuesta al comando *GATT_ReadUsingCharUUID* – Status: 0x00 (success)). Si la respuesta es errónea, se cesa la conexión entre el BLE y el Beacon y se comienza de nuevo el proceso. Si la respuesta no tiene errores, se sigue leyendo ésta hasta llegar a los datos 21 al 24, los cuales corresponden al valor de la temperatura, y se guardan en el array *resp_trama_temperatura*. Una vez obtenido el array *resp_trama_temperatura* se convierte este dato hexadecimal a decimal y ya se tiene el valor de la temperatura ambiente.

Finalmente, una vez terminada la lectura de la temperatura, se cesa la conexión entre el BLE y el Beacon de modo que éste último quede libre para poder ser utilizado por otros BLEs.

4.4 FUNCIÓN RELÉ

En los momentos en los que la temperatura medida esté en torno a la temperatura deseada, debido a que las mediciones fluctúan, la temperatura medida estará continuamente variando entre valores algo mayores y algo menores que la temperatura de consigna. En consecuencia, el actuador (en este caso el relé), estaría continuamente apangándose y encendiéndose. Para evitar eso, se introduce una histéresis en el control del relé, la cual evita ese encendido intermitente, dando lugar a un control más eficiente y disminuyendo las posibilidades de que el relé se estropee antes de lo que debiera.

El ciclo de histéresis se muestra en la siguiente imagen:

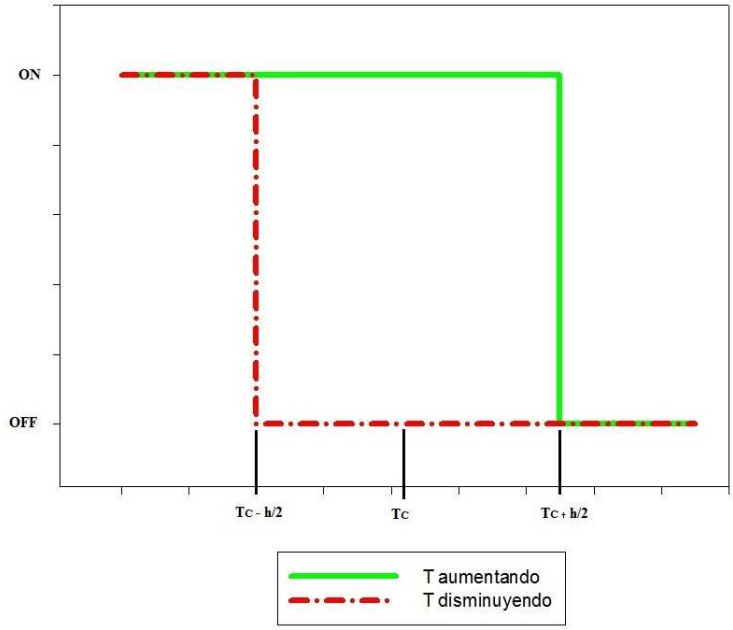


FIG. 64: CICLO DE HISTÉRESIS

En la imagen superior, la línea verde muestra el estado del actuador (en este caso el relé) cuando la temperatura está aumentando. Mientras la temperatura aumenta, la calefacción debe mantenerse encendida hasta superar el límite superior del ciclo de histéresis (T_C + h/2).

Cuando la temperatura se encuentre disminuyendo (línea roja), la calefacción no debe encenderse hasta el momento en el que se supere el límite inferior del ciclo de histéresis (T_C - h/2).

Así, la temperatura variará del siguiente modo:

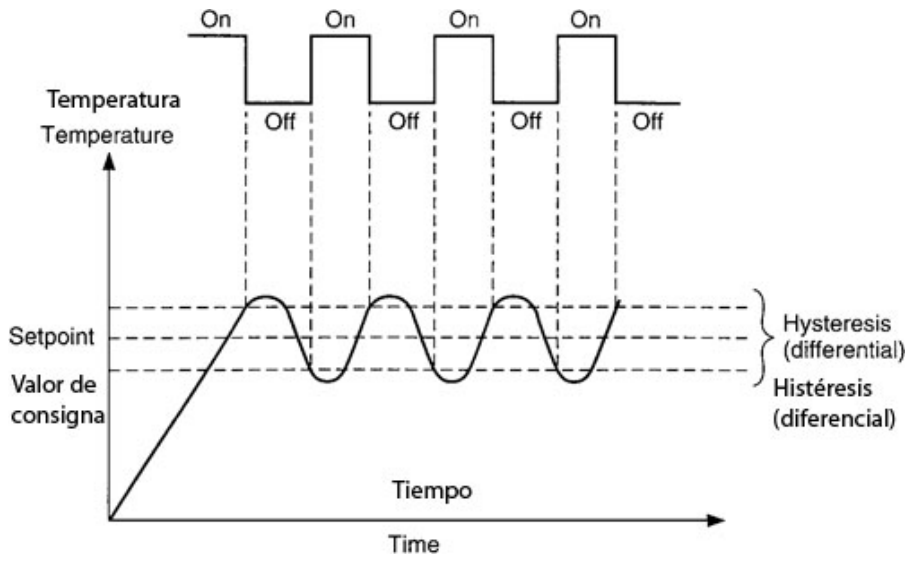


FIG. 65: EFECTO DE LA HISTÉRESIS EN LA TEMPERATURA

Cuando el actuador (relé) está encendido, la temperatura sube hasta superar el límite superior de la histéresis y a continuación el actuador se apaga haciendo que la temperatura descienda hasta sobrepasar el límite inferior. De este modo, se evita que el relé esté continuamente encendiéndose y apagándose cuando la temperatura medida se encuentra en torno a la temperatura de consigna.

Entonces, para el relé, se fija una histéresis de 1ºC; es decir, las temperaturas máxima y mínima del ciclo de histéresis serán 0.5ºC mayor y menor que la de consigna respectivamente. Así, cuando la temperatura medida es inferior al límite inferior de histéresis, entonces se enciende el relé, y se mantiene encendido hasta que la temperatura sobrepasa el límite superior de histéresis. Una vez apagado, sigue en este estado hasta llegar a temperaturas inferiores al límite inferior.

Para activar el relé se pone "HIGH" el pin de Arduino que lo controla (pin 2) ya que el relé funciona a nivel alto. Por el contrario, si se desea que el relé esté desactivado, se pone el pin 2 de Arduino "LOW".

4.5 FUNCIÓN OPTOACOPLADOR

Al igual que en el caso del relé, para el optoacoplador también se fija una histéresis de 1ºC; evitando así el continuo encendido y apagado del actuador en torno a la temperatura de consigna.

El control del optoacoplador es exactamente igual que el del relé. Para activarlo se pone el pin que lo controla (pin 4 de Arduino) en "HIGH" de modo que pasa corriente por el LED del optoacoplador, y el fototransistor entra en saturación. Para poner el transistor en corte, se pone en "LOW" el pin 4 y deja de pasar corriente por el LED.

4.6 FUNCIÓN MOTOR

De nuevo en este caso se emplea histéresis del mismo modo que se hace con el relé y el optoacoplador, evitando que el motor, y por tanto la válvula, se abra y cierre continuamente.

Si la calefacción debe estar encendida (si la temperatura está aumentando y es menor que el límite superior de histéresis), entonces es necesario abrir la válvula y dejar pasar agua a la caldera. Para ello, se recurre a la función "*abrirValvula*", la cual, tal y como se explica en el apartado 3.2.3, repite la secuencia 1-8 mostrada en la Fig. 48 128 veces para conseguir un giro de 90º en el motor.

Si, por el contrario, se desea que la calefacción esté apagada (la temperatura está disminuyendo y es mayor que el límite inferior de histéresis), entonces se recurre a la función "*cerrarValvula*" para cerrar la válvula y por tanto el paso de agua a la caldera. Esta función repite la secuencia 1-8 de manera inversa (8-1) 128 veces, de modo que se consigue el mismo giro que con la función "*abrirValvula*" pero en sentido contrario.

Los pines de Arduino empleados para el control del motor son los pines 6, 7, 8 y 9, correspondientes a los pines IN1, IN2, IN3 e IN4 del motor respectivamente.

4.7 FUNCIÓN ANALÓGICO

Esta función pretende implementar el controlador mostrado en la ecuación [21].

Una vez se tiene, a partir de dicha ecuación, el valor de la acción de control (u) deseado, se calcula el duty cycle necesario para obtener una señal PWM de valor medio igual a u.

$$u = \frac{t \cdot 5}{T} \Leftrightarrow t = \frac{u \cdot T}{5} \Rightarrow D = \frac{t}{T} \cdot 100(\%) \qquad \textbf{[29]}$$

Finalmente, mediante la función "*analogWrite()*" se genera la señal PWM deseada en el pin de control del módulo analógico. Este pin es el pin 3 de Arduino.

La acción integral tiene problemas cuando el controlador entra en saturación (cuando la acción de control necesaria es mayor que 5 o menor que 0). Cuando esto ocurre, la acción integral aumenta constantemente, y cuando el valor de la temperatura se acerca al deseado, el actuador no deja de saturar debido a que ha acumulado mucha acción integral, llegando a inestabilizar el sistema.

Para evitar esto, cuando el actuador entra en saturación, se deja de aumentar la parte integral de la acción de control; es decir, no se acumula la acción integral. Esto se puede observar en el software correspondiente a esta función: sólo se acumula acción integral cuando el valor de u necesario se encuentra entre 0 y 5V.

5. DEMOSTRACIÓN DE FUNCIONAMIENTO DEL PROTOTIPO

En el siguiente enlace se puede ver un vídeo en el que se muestra el funcionamiento de prototipo diseñado, utilizando un BLE como módulo sensor y un motor PAP como módulo actuador.

https://www.youtube.com/watch?v=y6pFpFNvv5k

Tal y como se explica en el vídeo, el funcionamiento del sistema es mucho más rápido (instantáneo) cuando el sensor empleado es otro distintos al BLE; ya que en el caso de éste sensor el tiempo necesario para la medición de la temperatura es muy elevado en comparación con las NTCs o los sensores de estado sólido.

6. PRESUPUESTO

En la siguiente tabla se muestra el precio final de cada uno de los módulos diseñados:

Componente	Cantidad	Precio unitario	Precio total	Precio total módulo
NTC				
Resistencia	5	0,02 €	0,10 €	**11,16 €**
NTC 1kΩ	1	0,915 €	0,92 €	
INA126PA	1	2,42 €	2,42 €	
Potenciómetro multivuelta	1	1,11 €	1,11 €	
TRA 1-0521	1	5,20 €	5,20 €	
Conector	3	0,47 €	1,41 €	
LM35				
LM35	1	4,96 €	4,96 €	**14,01 €**
Resistencia	1	0,02 €	0,02 €	
INA126PA	1	2,42 €	2,42 €	
TRA 1-0521	1	5,20 €	5,20 €	
Conector	3	0,47 €	1,41 €	
BLE y Beacon				
Beacon	1	19,01 €	19,01 €	**30,25 €**
Módulo BLE	1	9,50 €	9,50 €	
ZVN4424A	1	0,759 €	0,76 €	
Resistencia	2	0,02 €	0,04 €	
Conector	2	0,47 €	0,94 €	
Relé				
SRD-05VDC-SL-C	1	1,90 €	1,90 €	**2,84 €**
Conector	2	0,47 €	0,94 €	
Optoacoplador				
4N25	1	0,309 €	0,31 €	**1,29 €**
Resistencia	2	0,02 €	0,04 €	
Conector	2	0,47 €	0,94 €	
Motor PAP				
28BYJ-48	1	2,39 €	2,39 €	**4,63 €**
ULN2003	1	1,30 €	1,30 €	
Conector	2	0,47 €	0,94 €	
Control analógico				
Resistencia	3	0,02 €	0,06 €	**5,95 €**
Condensador	1	0,025 €	0,03 €	
UA741CN	1	0,288 €	0,29 €	
MCA05D12S	1	4,17 €	4,17 €	
Conector	3	0,47 €	1,41 €	

Componente	Cantidad	Precio unitario	Precio total	Precio total módulo
Controlador				
Arduino Uno	**1**	**20 €**	**20,00 €**	**40,00 €**
Ethernet Shield	1	20 €	20,00 €	
Precio total				**110,13 €**

El precio final total del prototipo es de 110.13€ que, aunque inicialmente pueda parecer elevado para un termostato, no lo es tanto si se tiene en cuenta que el objetivo fundamental de este proyecto era conseguir un sistema modular, tal y como se ha hecho. Por tanto, el usuario no deberá adquirir todos los módulos, sino únicamente el módulo sensor y el módulo actuador necesarios en cada caso; reduciéndose así considerablemente el precio del termostato.

Las posibles variantes sensor-actuador y el precio final del termostato para cada caso se recogen en la siguiente tabla, ordenadas de acuerdo al precio.

Combinación	Precio
NTC y optoacoplador	52,44 €
NTC y relé	54,00 €
LM35 y optoacoplador	55,30 €
NTC y motor PAP	55,79 €
LM35 y relé	56,85 €
NTC y control analógico	57,11 €
LM35 y motor PAP	58,64 €
LM35 y control analógico	59,96 €
BLE y optoacoplador	71,54 €
BLE y relé	73,09 €
BLE y motor PAP	74,88 €
BLE y control analógico	76,20 €

Como se observa en la tabla superior, el precio de todos los posibles termostatos ronda los 55€; exceptuando las opciones con módulo BLE, el cual encarece notablemente el sistema, pero también proporciona ventajas que la NTC y el LM35 no.

El precio actual de un termostato digital de gama media, sin opción de control remoto, ronda los 45€; por lo que el sistema obtenido resulta muy competitivo teniendo en cuenta la pequeña diferencia de precio respecto a los termostatos habituales en relación con los avances y las mayores posibilidades que ofrece.

7. LÍNEAS FUTURAS

En este proyecto, se ha conseguido desarrollar un sistema modular totalmente funcional. Sin embargo, aún se podría seguir desarrollando el prototipo añadiéndole módulos y características que le proporcionen más valor de cara al consumidor y lo hagan más atractivo:

- Incorporar una pantalla LCD que muestre la temperatura ambiente y la temperatura de consigna, así como un par de pulsadores que permitan modificar esta última directamente sobre el termostato, sin necesidad de acceder a la página web.
- Añadir un reloj de modo que se pueda implementar una opción de programación semanal de la calefacción, como ya hacen posible muchos de los termostatos actuales.
- Ajustar el tamaño de los distintos módulos, haciéndolos lo más pequeños posible, reduciendo así el tamaño total del termostato y haciéndolo más atractivo para el consumidor.
- Diseñar una carcasa que proporcione un buen aspecto y acabado al termostato
- Continuar desarrollando el software de modo que se pueda emplear más de un sensor, situados en distintas zonas de la casa, y controlar la calefacción de acuerdo a la temperatura media o al gradiente de temperatura.

8. CONCLUSIONES

Mediante la realización de este proyecto, por un lado, se han cumplido los objetivos inicialmente fijados para este proyecto:

- *Crear distintos módulos que funcionen independientemente pero sean capaces de funcionar en conjunto e interactuar a través del software, creando así un sistema modular en el que el usuario únicamente emplee los módulos necesarios:* se han implementado tres módulos de sensado así como cuatro de actuación, los cuales posibilitan la "creación" de 12 termostatos distintos de a cuerdo a las necesidades de cada hogar.
- *Conseguir un sistema que pueda ser manejado de manera remota a través de cualquier dispositivo con conexión a internet:* se ha creado una página web alojada en el servidor de Arduino de modo que el usuario pueda acceder a ella desde cualquier lugar y dispositivo y a través de la cual puede no sólo modificar el valor de la temperatura de consigna deseada sino también conocer la temperatura actual de consigna así como la temperatura ambiente y el estado (ON/OFF) de la calefacción en cada momento.

Por otro lado, se han puesto en práctica conocimientos adquiridos a lo largo de la carrera, y se han adquirido nuevos:

- Se han diseñado íntegramente sistemas tanto de sensado como de actuación, desde la selección de los componentes a emplear hasta el desarrollo de los circuitos necesarios en cada caso para obtener resultados óptimos. Por lo que ha sido necesario recurrir a conocimientos obtenidos en asignaturas como "Sensores y actuadores", "Fundamentos de electrónica" o "Sistemas electrónicos" entre otras.
- Se ha implementado un extenso software para el control de cada uno de los módulos; para cuyo desarrollo han sido de gran ayuda asignaturas como "Electrónica digital y microprocesadores" o "Ingeniería de control".
- Además, se han adquirido nuevos conocimientos básicos sobre la creación de páginas web en html y sobre el funcionamiento de los routers y servidores.
- También se ha aprendido sobre nuevas tecnologías como el BLE (Bluetooth Low Energy) y sobre distintos protocolos de comunicación como puede ser la comunicación serie o la comunicación SPI.

9. BIBLIOGRAFÍA

[1] http://calefaccion.quotatis.es/tipos-de-calefaccion

[2] http://en.wikipedia.org/wiki/Thermostat

[3] http://www.4ucontrol.com/tutoriales/gsm_esp.pdf

[4] http://arduino.cc/en/Guide/ArduinoYun

[5] http://arduino.cc/en/Main/ArduinoBoardYun?from=Products.ArduinoYUN

[6] http://www.ibeacon.com/what-is-ibeacon-a-guide-to-beacons/

[7] http://www.bluetooth.com/Pages/low-energy-tech-info.aspx

[8] The measurement, instrumentation and sensors handbook; John G. Webster; CRC Press & IEEE Press

[9] http://es.scribd.com/doc/97395385/Informe-de-Control-on-Off-Con-Histeresis

[10] http://arubia45.blogspot.com.es/2013/01/modulo-releerelay-arduino.html

[11] http://www.areatecnologia.com/electronica/optoacoplador.html

[12] http://www.todorobot.com.ar/tutorial-sobre-motores-paso-a-paso-stepper-motors/

[13]http://todoproductividad.blogspot.com.es/2009/10/motor-por-pasos-o-servomotor.html

[14]http://www.fenercom.com/pages/pdf/formacion/14-03-27_Jornada_emisores_eficientes/5_Regulacion_de_temperatura_HONEYWELL

[15] http://es.scribd.com/doc/94091423/Linealizacion-de-termistores#scribd

[16] http://docsetools.com/articulos-informativos/article_66380.html

[17] http://www.construmatica.com/construpedia/Acumulador

[18] http://www.hobbytronics.co.uk/mosfet-voltage-level-converter

[19] Rectificadores, tiristores y triacs; M. Gaudry; Ed. Paraninfo

[20] Power Electronics; M. J. Fisher; PWS-KENT

[21] Power electronics, circuits, devices and applications; H. Rashid Muhammad; Prentice Hall

[22] http://www.instructables.com/id/BYJ48-Stepper-Motor/?lang=es

[23] http://www.arduino.cc/en/Tutorial/PWM

[24] http://www.electroensaimada.com/pwm.html

ANEXOS

ANEXO 1: CÁLCULO DE LAS RESISTENCIAS R_1 Y R_2 PARA LA LINEALIZACIÓN DE UNA NTC DE 1K

R1	R2	RT01	BETA	To	Vref
9,95E+02	1,62E+03	1000	3730	298	4,45

*V_{ref} es 4.45V ya que Arduino no proporciona los 5V teóricos que debería, sino algo menos; en este caso 4.45V

T	RT	V1	Aproximación	Error relativo	I1	Intc
10	1941,45	2,0909471	2,10257	-0,56%	2,3709E-03	1,0770E-03
10,1	1932,43	2,0886	2,099893	-0,54%	2,3733E-03	1,0808E-03
10,2	1923,46	2,0862489	2,097216	-0,53%	2,3756E-03	1,0846E-03
10,3	1914,54	2,0838939	2,094539	-0,51%	2,3780E-03	1,0885E-03
10,4	1905,66	2,0815349	2,091862	-0,50%	2,3804E-03	1,0923E-03
10,5	1896,84	2,0791719	2,089185	-0,48%	2,3827E-03	1,0961E-03
10,6	1888,06	2,076805	2,086508	-0,47%	2,3851E-03	1,1000E-03
10,7	1879,33	2,0744342	2,083831	-0,45%	2,3875E-03	1,1038E-03
10,8	1870,64	2,0720595	2,081154	-0,44%	2,3899E-03	1,1077E-03
10,9	1862	2,069681	2,078477	-0,42%	2,3923E-03	1,1115E-03
11	1853,4	2,0672985	2,0758	-0,41%	2,3947E-03	1,1154E-03
11,1	1844,86	2,0649123	2,073123	-0,40%	2,3971E-03	1,1193E-03
11,2	1836,35	2,0625222	2,070446	-0,38%	2,3995E-03	1,1232E-03
11,3	1827,9	2,0601282	2,067769	-0,37%	2,4019E-03	1,1270E-03
11,4	1819,48	2,0577305	2,065092	-0,36%	2,4043E-03	1,1309E-03
11,5	1811,11	2,055329	2,062415	-0,34%	2,4067E-03	1,1348E-03
11,6	1802,79	2,0529237	2,059738	-0,33%	2,4091E-03	1,1387E-03
11,7	1794,51	2,0505147	2,057061	-0,32%	2,4115E-03	1,1427E-03
11,8	1786,27	2,048102	2,054384	-0,31%	2,4140E-03	1,1466E-03
11,9	1778,08	2,0456855	2,051707	-0,29%	2,4164E-03	1,1505E-03
12	1769,93	2,0432653	2,04903	-0,28%	2,4188E-03	1,1544E-03
12,1	1761,82	2,0408415	2,046353	-0,27%	2,4213E-03	1,1584E-03
12,2	1753,76	2,038414	2,043676	-0,26%	2,4237E-03	1,1623E-03
12,3	1745,74	2,0359828	2,040999	-0,25%	2,4261E-03	1,1663E-03
12,4	1737,76	2,033548	2,038322	-0,23%	2,4286E-03	1,1702E-03
12,5	1729,82	2,0311096	2,035645	-0,22%	2,4310E-03	1,1742E-03
12,6	1721,93	2,0286677	2,032968	-0,21%	2,4335E-03	1,1781E-03
12,7	1714,08	2,0262221	2,030291	-0,20%	2,4360E-03	1,1821E-03
12,8	1706,26	2,023773	2,027614	-0,19%	2,4384E-03	1,1861E-03
12,9	1698,49	2,0213203	2,024937	-0,18%	2,4409E-03	1,1901E-03
13	1690,76	2,0188641	2,02226	-0,17%	2,4434E-03	1,1941E-03
13,1	1683,07	2,0164044	2,019583	-0,16%	2,4458E-03	1,1980E-03
13,2	1675,42	2,0139413	2,016906	-0,15%	2,4483E-03	1,2020E-03

13,3	1667,81	2,0114746	2,014229	-0,14%	2,4508E-03	1,2061E-03
13,4	1660,24	2,0090045	2,011552	-0,13%	2,4533E-03	1,2101E-03
13,5	1652,71	2,006531	2,008875	-0,12%	2,4557E-03	1,2141E-03
13,6	1645,22	2,0040541	2,006198	-0,11%	2,4582E-03	1,2181E-03
13,7	1637,77	2,0015737	2,003521	-0,10%	2,4607E-03	1,2221E-03
13,8	1630,36	1,99909	2,000844	-0,09%	2,4632E-03	1,2262E-03
13,9	1622,99	1,9966029	1,998167	-0,08%	2,4657E-03	1,2302E-03
14	1615,65	1,9941125	1,99549	-0,07%	2,4682E-03	1,2342E-03
14,1	1608,35	1,9916188	1,992813	-0,06%	2,4707E-03	1,2383E-03
14,2	1601,09	1,9891218	1,990136	-0,05%	2,4732E-03	1,2424E-03
14,3	1593,87	1,9866215	1,987459	-0,04%	2,4758E-03	1,2464E-03
14,4	1586,69	1,9841179	1,984782	-0,03%	2,4783E-03	1,2505E-03
14,5	1579,54	1,9816111	1,982105	-0,02%	2,4808E-03	1,2545E-03
14,6	1572,43	1,979101	1,979428	-0,02%	2,4833E-03	1,2586E-03
14,7	1565,36	1,9765878	1,976751	-0,01%	2,4858E-03	1,2627E-03
14,8	1558,32	1,9740714	1,974074	0,00%	2,4884E-03	1,2668E-03
14,9	1551,32	1,9715518	1,971397	0,01%	2,4909E-03	1,2709E-03
15	1544,36	1,969029	1,96872	0,02%	2,4934E-03	1,2750E-03
15,1	1537,43	1,9665032	1,966043	0,02%	2,4960E-03	1,2791E-03
15,2	1530,54	1,9639742	1,963366	0,03%	2,4985E-03	1,2832E-03
15,3	1523,69	1,9614421	1,960689	0,04%	2,5011E-03	1,2873E-03
15,4	1516,87	1,958907	1,958012	0,05%	2,5036E-03	1,2914E-03
15,5	1510,08	1,9563688	1,955335	0,05%	2,5062E-03	1,2955E-03
15,6	1503,33	1,9538276	1,952658	0,06%	2,5087E-03	1,2997E-03
15,7	1496,62	1,9512834	1,949981	0,07%	2,5113E-03	1,3038E-03
15,8	1489,94	1,9487362	1,947304	0,07%	2,5138E-03	1,3079E-03
15,9	1483,29	1,9461861	1,944627	0,08%	2,5164E-03	1,3121E-03
16	1476,68	1,9436329	1,94195	0,09%	2,5190E-03	1,3162E-03
16,1	1470,1	1,9410769	1,939273	0,09%	2,5215E-03	1,3204E-03
16,2	1463,56	1,938518	1,936596	0,10%	2,5241E-03	1,3245E-03
16,3	1457,05	1,9359562	1,933919	0,11%	2,5267E-03	1,3287E-03
16,4	1450,57	1,9333915	1,931242	0,11%	2,5293E-03	1,3328E-03
16,5	1444,13	1,9308239	1,928565	0,12%	2,5318E-03	1,3370E-03
16,6	1437,72	1,9282536	1,925888	0,12%	2,5344E-03	1,3412E-03
16,7	1431,34	1,9256804	1,923211	0,13%	2,5370E-03	1,3454E-03
16,8	1424,99	1,9231045	1,920534	0,13%	2,5396E-03	1,3496E-03
16,9	1418,68	1,9205258	1,917857	0,14%	2,5422E-03	1,3537E-03
17	1412,4	1,9179444	1,91518	0,14%	2,5448E-03	1,3579E-03
17,1	1406,15	1,9153602	1,912503	0,15%	2,5474E-03	1,3621E-03
17,2	1399,94	1,9127734	1,909826	0,15%	2,5500E-03	1,3663E-03
17,3	1393,75	1,9101838	1,907149	0,16%	2,5526E-03	1,3705E-03
17,4	1387,6	1,9075917	1,904472	0,16%	2,5552E-03	1,3747E-03
17,5	1381,48	1,9049969	1,901795	0,17%	2,5578E-03	1,3790E-03
17,6	1375,39	1,9023994	1,899118	0,17%	2,5604E-03	1,3832E-03
17,7	1369,33	1,8997994	1,896441	0,18%	2,5630E-03	1,3874E-03
17,8	1363,3	1,8971968	1,893764	0,18%	2,5656E-03	1,3916E-03

17,9	1357,3	1,8945917	1,891087	0,18%	2,5682E-03	1,3959E-03
18	1351,33	1,8919841	1,88841	0,19%	2,5709E-03	1,4001E-03
18,1	1345,39	1,8893739	1,885733	0,19%	2,5735E-03	1,4043E-03
18,2	1339,49	1,8867613	1,883056	0,20%	2,5761E-03	1,4086E-03
18,3	1333,61	1,8841462	1,880379	0,20%	2,5787E-03	1,4128E-03
18,4	1327,76	1,8815286	1,877702	0,20%	2,5814E-03	1,4171E-03
18,5	1321,95	1,8789087	1,875025	0,21%	2,5840E-03	1,4213E-03
18,6	1316,16	1,8762863	1,872348	0,21%	2,5866E-03	1,4256E-03
18,7	1310,4	1,8736616	1,869671	0,21%	2,5893E-03	1,4298E-03
18,8	1304,67	1,8710345	1,866994	0,22%	2,5919E-03	1,4341E-03
18,9	1298,97	1,8684051	1,864317	0,22%	2,5946E-03	1,4384E-03
19	1293,3	1,8657734	1,86164	0,22%	2,5972E-03	1,4426E-03
19,1	1287,65	1,8631394	1,858963	0,22%	2,5999E-03	1,4469E-03
19,2	1282,04	1,8605032	1,856286	0,23%	2,6025E-03	1,4512E-03
19,3	1276,45	1,8578647	1,853609	0,23%	2,6052E-03	1,4555E-03
19,4	1270,89	1,855224	1,850932	0,23%	2,6078E-03	1,4598E-03
19,5	1265,36	1,8525811	1,848255	0,23%	2,6105E-03	1,4641E-03
19,6	1259,86	1,849936	1,845578	0,24%	2,6131E-03	1,4684E-03
19,7	1254,38	1,8472888	1,842901	0,24%	2,6158E-03	1,4727E-03
19,8	1248,94	1,8446395	1,840224	0,24%	2,6185E-03	1,4770E-03
19,9	1243,52	1,841988	1,837547	0,24%	2,6211E-03	1,4813E-03
20	1238,12	1,8393345	1,83487	0,24%	2,6238E-03	1,4856E-03
20,1	1232,76	1,8366789	1,832193	0,24%	2,6265E-03	1,4899E-03
20,2	1227,42	1,8340212	1,829516	0,25%	2,6291E-03	1,4942E-03
20,3	1222,11	1,8313616	1,826839	0,25%	2,6318E-03	1,4985E-03
20,4	1216,82	1,8287	1,824162	0,25%	2,6345E-03	1,5029E-03
20,5	1211,56	1,8260364	1,821485	0,25%	2,6371E-03	1,5072E-03
20,6	1206,33	1,8233708	1,818808	0,25%	2,6398E-03	1,5115E-03
20,7	1201,12	1,8207033	1,816131	0,25%	2,6425E-03	1,5158E-03
20,8	1195,94	1,818034	1,813454	0,25%	2,6452E-03	1,5202E-03
20,9	1190,79	1,8153627	1,810777	0,25%	2,6479E-03	1,5245E-03
21	1185,66	1,8126896	1,8081	0,25%	2,6506E-03	1,5288E-03
21,1	1180,55	1,8100147	1,805423	0,25%	2,6533E-03	1,5332E-03
21,2	1175,47	1,8073379	1,802746	0,25%	2,6559E-03	1,5375E-03
21,3	1170,42	1,8046594	1,800069	0,25%	2,6586E-03	1,5419E-03
21,4	1165,39	1,8019791	1,797392	0,25%	2,6613E-03	1,5462E-03
21,5	1160,39	1,7992971	1,794715	0,25%	2,6640E-03	1,5506E-03
21,6	1155,41	1,7966133	1,792038	0,25%	2,6667E-03	1,5550E-03
21,7	1150,46	1,7939279	1,789361	0,25%	2,6694E-03	1,5593E-03
21,8	1145,53	1,7912408	1,786684	0,25%	2,6721E-03	1,5637E-03
21,9	1140,63	1,788552	1,784007	0,25%	2,6748E-03	1,5680E-03
22	1135,75	1,7858617	1,78133	0,25%	2,6775E-03	1,5724E-03
22,1	1130,89	1,7831697	1,778653	0,25%	2,6802E-03	1,5768E-03
22,2	1126,06	1,7804761	1,775976	0,25%	2,6829E-03	1,5812E-03
22,3	1121,25	1,777781	1,773299	0,25%	2,6856E-03	1,5855E-03
22,4	1116,47	1,7750844	1,770622	0,25%	2,6884E-03	1,5899E-03

22,5	1111,71	1,7723863	1,767945	0,25%	2,6911E-03	1,5943E-03
22,6	1106,97	1,7696866	1,765268	0,25%	2,6938E-03	1,5987E-03
22,7	1102,25	1,7669856	1,762591	0,25%	2,6965E-03	1,6031E-03
22,8	1097,56	1,764283	1,759914	0,25%	2,6992E-03	1,6075E-03
22,9	1092,9	1,7615791	1,757237	0,25%	2,7019E-03	1,6118E-03
23	1088,25	1,7588738	1,75456	0,25%	2,7046E-03	1,6162E-03
23,1	1083,63	1,7561671	1,751883	0,24%	2,7074E-03	1,6206E-03
23,2	1079,03	1,7534591	1,749206	0,24%	2,7101E-03	1,6250E-03
23,3	1074,46	1,7507497	1,746529	0,24%	2,7128E-03	1,6294E-03
23,4	1069,9	1,7480391	1,743852	0,24%	2,7155E-03	1,6338E-03
23,5	1065,37	1,7453272	1,741175	0,24%	2,7183E-03	1,6382E-03
23,6	1060,86	1,742614	1,738498	0,24%	2,7210E-03	1,6426E-03
23,7	1056,37	1,7398997	1,735821	0,23%	2,7237E-03	1,6470E-03
23,8	1051,91	1,7371841	1,733144	0,23%	2,7264E-03	1,6515E-03
23,9	1047,47	1,7344673	1,730467	0,23%	2,7292E-03	1,6559E-03
24	1043,04	1,7317494	1,72779	0,23%	2,7319E-03	1,6603E-03
24,1	1038,64	1,7290304	1,725113	0,23%	2,7346E-03	1,6647E-03
24,2	1034,27	1,7263102	1,722436	0,22%	2,7374E-03	1,6691E-03
24,3	1029,91	1,723589	1,719759	0,22%	2,7401E-03	1,6735E-03
24,4	1025,57	1,7208667	1,717082	0,22%	2,7428E-03	1,6780E-03
24,5	1021,26	1,7181434	1,714405	0,22%	2,7456E-03	1,6824E-03
24,6	1016,97	1,715419	1,711728	0,22%	2,7483E-03	1,6868E-03
24,7	1012,69	1,7126937	1,709051	0,21%	2,7511E-03	1,6912E-03
24,8	1008,44	1,7099674	1,706374	0,21%	2,7538E-03	1,6957E-03
24,9	1004,21	1,7072402	1,703697	0,21%	2,7565E-03	1,7001E-03
25	1000	1,7045121	1,70102	0,20%	2,7593E-03	1,7045E-03
25,1	995,81	1,701783	1,698343	0,20%	2,7620E-03	1,7089E-03
25,2	991,64	1,6990531	1,695666	0,20%	2,7648E-03	1,7134E-03
25,3	987,491	1,6963223	1,692989	0,20%	2,7675E-03	1,7178E-03
25,4	983,361	1,6935908	1,690312	0,19%	2,7703E-03	1,7222E-03
25,5	979,252	1,6908584	1,687635	0,19%	2,7730E-03	1,7267E-03
25,6	975,163	1,6881252	1,684958	0,19%	2,7758E-03	1,7311E-03
25,7	971,093	1,6853913	1,682281	0,18%	2,7785E-03	1,7356E-03
25,8	967,043	1,6826567	1,679604	0,18%	2,7812E-03	1,7400E-03
25,9	963,013	1,6799214	1,676927	0,18%	2,7840E-03	1,7444E-03
26	959,002	1,6771853	1,67425	0,18%	2,7867E-03	1,7489E-03
26,1	955,01	1,6744487	1,671573	0,17%	2,7895E-03	1,7533E-03
26,2	951,038	1,6717113	1,668896	0,17%	2,7922E-03	1,7578E-03
26,3	947,085	1,6689734	1,666219	0,17%	2,7950E-03	1,7622E-03
26,4	943,151	1,6662349	1,663542	0,16%	2,7978E-03	1,7667E-03
26,5	939,236	1,6634958	1,660865	0,16%	2,8005E-03	1,7711E-03
26,6	935,34	1,6607562	1,658188	0,15%	2,8033E-03	1,7756E-03
26,7	931,462	1,6580161	1,655511	0,15%	2,8060E-03	1,7800E-03
26,8	927,604	1,6552755	1,652834	0,15%	2,8088E-03	1,7845E-03
26,9	923,763	1,6525344	1,650157	0,14%	2,8115E-03	1,7889E-03
27	919,942	1,6497929	1,64748	0,14%	2,8143E-03	1,7934E-03

27,1	916,138	1,6470509	1,644803	0,14%	2,8170E-03	1,7978E-03
27,2	912,353	1,6443085	1,642126	0,13%	2,8198E-03	1,8023E-03
27,3	908,586	1,6415658	1,639449	0,13%	2,8225E-03	1,8067E-03
27,4	904,837	1,6388227	1,636772	0,13%	2,8253E-03	1,8112E-03
27,5	901,106	1,6360793	1,634095	0,12%	2,8281E-03	1,8156E-03
27,6	897,392	1,6333356	1,631418	0,12%	2,8308E-03	1,8201E-03
27,7	893,697	1,6305916	1,628741	0,11%	2,8336E-03	1,8245E-03
27,8	890,019	1,6278473	1,626064	0,11%	2,8363E-03	1,8290E-03
27,9	886,359	1,6251028	1,623387	0,11%	2,8391E-03	1,8335E-03
28	882,716	1,6223581	1,62071	0,10%	2,8419E-03	1,8379E-03
28,1	879,09	1,6196132	1,618033	0,10%	2,8446E-03	1,8424E-03
28,2	875,482	1,6168682	1,615356	0,09%	2,8474E-03	1,8468E-03
28,3	871,891	1,614123	1,612679	0,09%	2,8501E-03	1,8513E-03
28,4	868,317	1,6113776	1,610002	0,09%	2,8529E-03	1,8557E-03
28,5	864,761	1,6086322	1,607325	0,08%	2,8556E-03	1,8602E-03
28,6	861,221	1,6058867	1,604648	0,08%	2,8584E-03	1,8647E-03
28,7	857,698	1,6031412	1,601971	0,07%	2,8612E-03	1,8691E-03
28,8	854,191	1,6003956	1,599294	0,07%	2,8639E-03	1,8736E-03
28,9	850,701	1,5976501	1,596617	0,06%	2,8667E-03	1,8780E-03
29	847,228	1,5949045	1,59394	0,06%	2,8694E-03	1,8825E-03
29,1	843,772	1,592159	1,591263	0,06%	2,8722E-03	1,8870E-03
29,2	840,331	1,5894135	1,588586	0,05%	2,8750E-03	1,8914E-03
29,3	836,907	1,5866682	1,585909	0,05%	2,8777E-03	1,8959E-03
29,4	833,499	1,5839229	1,583232	0,04%	2,8805E-03	1,9003E-03
29,5	830,108	1,5811778	1,580555	0,04%	2,8832E-03	1,9048E-03
29,6	826,732	1,5784329	1,577878	0,04%	2,8860E-03	1,9092E-03
29,7	823,372	1,5756881	1,575201	0,03%	2,8888E-03	1,9137E-03
29,8	820,028	1,5729435	1,572524	0,03%	2,8915E-03	1,9182E-03
29,9	816,7	1,5701992	1,569847	0,02%	2,8943E-03	1,9226E-03
30	813,388	1,5674551	1,56717	0,02%	2,8970E-03	1,9271E-03
30,1	810,091	1,5647113	1,564493	0,01%	2,8998E-03	1,9315E-03
30,2	806,809	1,5619677	1,561816	0,01%	2,9025E-03	1,9360E-03
30,3	803,544	1,5592245	1,559139	0,01%	2,9053E-03	1,9404E-03
30,4	800,293	1,5564816	1,556462	0,00%	2,9081E-03	1,9449E-03
30,5	797,058	1,5537391	1,553785	0,00%	2,9108E-03	1,9493E-03
30,6	793,838	1,5509969	1,551108	-0,01%	2,9136E-03	1,9538E-03
30,7	790,633	1,5482552	1,548431	-0,01%	2,9163E-03	1,9582E-03
30,8	787,443	1,5455139	1,545754	-0,02%	2,9191E-03	1,9627E-03
30,9	784,268	1,542773	1,543077	-0,02%	2,9218E-03	1,9671E-03
31	781,108	1,5400326	1,5404	-0,02%	2,9246E-03	1,9716E-03
31,1	777,963	1,5372928	1,537723	-0,03%	2,9273E-03	1,9760E-03
31,2	774,832	1,5345534	1,535046	-0,03%	2,9301E-03	1,9805E-03
31,3	771,716	1,5318145	1,532369	-0,04%	2,9328E-03	1,9849E-03
31,4	768,615	1,5290763	1,529692	-0,04%	2,9356E-03	1,9894E-03
31,5	765,528	1,5263386	1,527015	-0,04%	2,9384E-03	1,9938E-03
31,6	762,456	1,5236015	1,524338	-0,05%	2,9411E-03	1,9983E-03

31,7	759,398	1,520865	1,521661	-0,05%	2,9439E-03	2,0027E-03
31,8	756,354	1,5181292	1,518984	-0,06%	2,9466E-03	2,0072E-03
31,9	753,324	1,515394	1,516307	-0,06%	2,9494E-03	2,0116E-03
32	750,309	1,5126596	1,51363	-0,06%	2,9521E-03	2,0160E-03
32,1	747,307	1,5099259	1,510953	-0,07%	2,9548E-03	2,0205E-03
32,2	744,32	1,5071928	1,508276	-0,07%	2,9576E-03	2,0249E-03
32,3	741,346	1,5044606	1,505599	-0,08%	2,9603E-03	2,0294E-03
32,4	738,386	1,5017291	1,502922	-0,08%	2,9631E-03	2,0338E-03
32,5	735,44	1,4989985	1,500245	-0,08%	2,9658E-03	2,0382E-03
32,6	732,508	1,4962686	1,497568	-0,09%	2,9686E-03	2,0427E-03
32,7	729,589	1,4935396	1,494891	-0,09%	2,9713E-03	2,0471E-03
32,8	726,684	1,4908115	1,492214	-0,09%	2,9741E-03	2,0515E-03
32,9	723,792	1,4880842	1,489537	-0,10%	2,9768E-03	2,0560E-03
33	720,914	1,4853579	1,48686	-0,10%	2,9795E-03	2,0604E-03
33,1	718,048	1,4826324	1,484183	-0,10%	2,9823E-03	2,0648E-03
33,2	715,197	1,479908	1,481506	-0,11%	2,9850E-03	2,0692E-03
33,3	712,358	1,4771845	1,478829	-0,11%	2,9878E-03	2,0737E-03
33,4	709,532	1,474462	1,476152	-0,11%	2,9905E-03	2,0781E-03
33,5	706,72	1,4717405	1,473475	-0,12%	2,9932E-03	2,0825E-03
33,6	703,92	1,46902	1,470798	-0,12%	2,9960E-03	2,0869E-03
33,7	701,133	1,4663006	1,468121	-0,12%	2,9987E-03	2,0913E-03
33,8	698,36	1,4635822	1,465444	-0,13%	3,0014E-03	2,0957E-03
33,9	695,599	1,460865	1,462767	-0,13%	3,0042E-03	2,1002E-03
34	692,85	1,4581489	1,46009	-0,13%	3,0069E-03	2,1046E-03
34,1	690,115	1,4554339	1,457413	-0,14%	3,0096E-03	2,1090E-03
34,2	687,391	1,4527201	1,454736	-0,14%	3,0123E-03	2,1134E-03
34,3	684,681	1,4500074	1,452059	-0,14%	3,0151E-03	2,1178E-03
34,4	681,983	1,4472959	1,449382	-0,14%	3,0178E-03	2,1222E-03
34,5	679,297	1,4445857	1,446705	-0,15%	3,0205E-03	2,1266E-03
34,6	676,623	1,4418767	1,444028	-0,15%	3,0232E-03	2,1310E-03
34,7	673,962	1,439169	1,441351	-0,15%	3,0260E-03	2,1354E-03
34,8	671,313	1,4364625	1,438674	-0,15%	3,0287E-03	2,1398E-03
34,9	668,676	1,4337573	1,435997	-0,16%	3,0314E-03	2,1442E-03
35	666,051	1,4310535	1,43332	-0,16%	3,0341E-03	2,1486E-03
35,1	663,438	1,428351	1,430643	-0,16%	3,0368E-03	2,1530E-03
35,2	660,837	1,4256498	1,427966	-0,16%	3,0395E-03	2,1573E-03
35,3	658,248	1,4229501	1,425289	-0,16%	3,0423E-03	2,1617E-03
35,4	655,671	1,4202517	1,422612	-0,17%	3,0450E-03	2,1661E-03
35,5	653,105	1,4175548	1,419935	-0,17%	3,0477E-03	2,1705E-03
35,6	650,552	1,4148592	1,417258	-0,17%	3,0504E-03	2,1749E-03
35,7	648,009	1,4121652	1,414581	-0,17%	3,0531E-03	2,1792E-03
35,8	645,479	1,4094726	1,411904	-0,17%	3,0558E-03	2,1836E-03
35,9	642,96	1,4067815	1,409227	-0,17%	3,0585E-03	2,1880E-03
36	640,452	1,4040919	1,40655	-0,18%	3,0612E-03	2,1923E-03
36,1	637,956	1,4014039	1,403873	-0,18%	3,0639E-03	2,1967E-03
36,2	635,471	1,3987174	1,401196	-0,18%	3,0666E-03	2,2011E-03

36,3	632,997	1,3960325	1,398519	-0,18%	3,0693E-03	2,2054E-03
36,4	630,535	1,3933492	1,395842	-0,18%	3,0720E-03	2,2098E-03
36,5	628,083	1,3906675	1,393165	-0,18%	3,0747E-03	2,2141E-03
36,6	625,643	1,3879874	1,390488	-0,18%	3,0774E-03	2,2185E-03
36,7	623,214	1,385309	1,387811	-0,18%	3,0801E-03	2,2228E-03
36,8	620,796	1,3826322	1,385134	-0,18%	3,0828E-03	2,2272E-03
36,9	618,389	1,3799572	1,382457	-0,18%	3,0855E-03	2,2315E-03
37	615,993	1,3772838	1,37978	-0,18%	3,0882E-03	2,2359E-03
37,1	613,607	1,3746121	1,377103	-0,18%	3,0908E-03	2,2402E-03
37,2	611,232	1,3719422	1,374426	-0,18%	3,0935E-03	2,2446E-03
37,3	608,868	1,3692741	1,371749	-0,18%	3,0962E-03	2,2489E-03
37,4	606,515	1,3666077	1,369072	-0,18%	3,0989E-03	2,2532E-03
37,5	604,172	1,3639431	1,366395	-0,18%	3,1016E-03	2,2575E-03
37,6	601,84	1,3612804	1,363718	-0,18%	3,1042E-03	2,2619E-03
37,7	599,518	1,3586194	1,361041	-0,18%	3,1069E-03	2,2662E-03
37,8	597,207	1,3559604	1,358364	-0,18%	3,1096E-03	2,2705E-03
37,9	594,906	1,3533032	1,355687	-0,18%	3,1123E-03	2,2748E-03
38	592,616	1,3506478	1,35301	-0,17%	3,1149E-03	2,2791E-03
38,1	590,335	1,3479944	1,350333	-0,17%	3,1176E-03	2,2834E-03
38,2	588,065	1,3453429	1,347656	-0,17%	3,1203E-03	2,2877E-03
38,3	585,805	1,3426933	1,344979	-0,17%	3,1229E-03	2,2920E-03
38,4	583,556	1,3400457	1,342302	-0,17%	3,1256E-03	2,2963E-03
38,5	581,316	1,3374001	1,339625	-0,17%	3,1282E-03	2,3006E-03
38,6	579,086	1,3347565	1,336948	-0,16%	3,1309E-03	2,3049E-03
38,7	576,867	1,3321148	1,334271	-0,16%	3,1336E-03	2,3092E-03
38,8	574,657	1,3294752	1,331594	-0,16%	3,1362E-03	2,3135E-03
38,9	572,457	1,3268377	1,328917	-0,16%	3,1389E-03	2,3178E-03
39	570,267	1,3242022	1,32624	-0,15%	3,1415E-03	2,3221E-03
39,1	568,087	1,3215687	1,323563	-0,15%	3,1442E-03	2,3263E-03
39,2	565,916	1,3189374	1,320886	-0,15%	3,1468E-03	2,3306E-03
39,3	563,756	1,3163082	1,318209	-0,14%	3,1494E-03	2,3349E-03
39,4	561,604	1,3136811	1,315532	-0,14%	3,1521E-03	2,3392E-03
39,5	559,463	1,3110561	1,312855	-0,14%	3,1547E-03	2,3434E-03
39,6	557,331	1,3084333	1,310178	-0,13%	3,1574E-03	2,3477E-03
39,7	555,208	1,3058127	1,307501	-0,13%	3,1600E-03	2,3519E-03
39,8	553,095	1,3031943	1,304824	-0,13%	3,1626E-03	2,3562E-03
39,9	550,991	1,3005781	1,302147	-0,12%	3,1652E-03	2,3604E-03
40	548,896	1,2979641	1,29947	-0,12%	3,1679E-03	2,3647E-03

ANEXO 2: CÁLCULO DE LAS RESISTENCIAS R_1 Y R_2 PARA LA LINEALIZACIÓN DE UNA NTC DE 10K

R1	R2	RT01	BETA	To	Vref
9,95E+03	1,49E+04	10000	3977	298	4,45

*V_{ref} es 4.45V ya que Arduino no proporciona los 5V teóricos que debería, sino algo menos; en este caso 4.45V

T	RT	V1	Aproximación	Error relativo	I1	Intc
10	20286,4	2,0635542	2,07858	-0,73%	2,3984E-04	1,0172E-04
10,1	20186	2,0612212	2,075839	-0,71%	2,4008E-04	1,0211E-04
10,2	20086,1	2,0588836	2,073098	-0,69%	2,4031E-04	1,0250E-04
10,3	19986,8	2,0565414	2,070357	-0,67%	2,4055E-04	1,0290E-04
10,4	19888	2,0541946	2,067616	-0,65%	2,4078E-04	1,0329E-04
10,5	19789,8	2,0518431	2,064875	-0,64%	2,4102E-04	1,0368E-04
10,6	19692,1	2,0494871	2,062134	-0,62%	2,4126E-04	1,0408E-04
10,7	19595	2,0471265	2,059393	-0,60%	2,4149E-04	1,0447E-04
10,8	19498,5	2,0447613	2,056652	-0,58%	2,4173E-04	1,0487E-04
10,9	19402,5	2,0423916	2,053911	-0,56%	2,4197E-04	1,0526E-04
11	19307	2,0400174	2,05117	-0,55%	2,4221E-04	1,0566E-04
11,1	19212,1	2,0376386	2,048429	-0,53%	2,4245E-04	1,0606E-04
11,2	19117,7	2,0352554	2,045688	-0,51%	2,4269E-04	1,0646E-04
11,3	19023,8	2,0328677	2,042947	-0,50%	2,4293E-04	1,0686E-04
11,4	18930,5	2,0304756	2,040206	-0,48%	2,4317E-04	1,0726E-04
11,5	18837,7	2,028079	2,037465	-0,46%	2,4341E-04	1,0766E-04
11,6	18745,4	2,0256779	2,034724	-0,45%	2,4365E-04	1,0806E-04
11,7	18653,6	2,0232725	2,031983	-0,43%	2,4389E-04	1,0847E-04
11,8	18562,3	2,0208627	2,029242	-0,41%	2,4413E-04	1,0887E-04
11,9	18471,5	2,0184485	2,026501	-0,40%	2,4438E-04	1,0927E-04
12	18381,3	2,0160299	2,02376	-0,38%	2,4462E-04	1,0968E-04
12,1	18291,5	2,013607	2,021019	-0,37%	2,4486E-04	1,1008E-04
12,2	18202,3	2,0111798	2,018278	-0,35%	2,4511E-04	1,1049E-04
12,3	18113,5	2,0087483	2,015537	-0,34%	2,4535E-04	1,1090E-04
12,4	18025,3	2,0063124	2,012796	-0,32%	2,4560E-04	1,1131E-04
12,5	17937,5	2,0038723	2,010055	-0,31%	2,4584E-04	1,1171E-04
12,6	17850,3	2,001428	2,007314	-0,29%	2,4609E-04	1,1212E-04
12,7	17763,5	1,9989794	2,004573	-0,28%	2,4633E-04	1,1253E-04
12,8	17677,2	1,9965266	2,001832	-0,27%	2,4658E-04	1,1294E-04
12,9	17591,3	1,9940697	1,999091	-0,25%	2,4683E-04	1,1336E-04
13	17506	1,9916085	1,99635	-0,24%	2,4707E-04	1,1377E-04
13,1	17421,1	1,9891432	1,993609	-0,22%	2,4732E-04	1,1418E-04
13,2	17336,7	1,9866737	1,990868	-0,21%	2,4757E-04	1,1459E-04
13,3	17252,7	1,9842001	1,988127	-0,20%	2,4782E-04	1,1501E-04
13,4	17169,3	1,9817224	1,985386	-0,18%	2,4807E-04	1,1542E-04

13,5	17086,2	1,9792406	1,982645	-0,17%	2,4832E-04	1,1584E-04
13,6	17003,7	1,9767548	1,979904	-0,16%	2,4857E-04	1,1625E-04
13,7	16921,6	1,9742649	1,977163	-0,15%	2,4882E-04	1,1667E-04
13,8	16839,9	1,971771	1,974422	-0,13%	2,4907E-04	1,1709E-04
13,9	16758,7	1,969273	1,971681	-0,12%	2,4932E-04	1,1751E-04
14	16678	1,9667711	1,96894	-0,11%	2,4957E-04	1,1793E-04
14,1	16597,7	1,9642652	1,966199	-0,10%	2,4982E-04	1,1835E-04
14,2	16517,8	1,9617554	1,963458	-0,09%	2,5007E-04	1,1877E-04
14,3	16438,4	1,9592416	1,960717	-0,08%	2,5033E-04	1,1919E-04
14,4	16359,4	1,9567239	1,957976	-0,06%	2,5058E-04	1,1961E-04
14,5	16280,9	1,9542024	1,955235	-0,05%	2,5083E-04	1,2003E-04
14,6	16202,8	1,9516769	1,952494	-0,04%	2,5109E-04	1,2045E-04
14,7	16125,1	1,9491476	1,949753	-0,03%	2,5134E-04	1,2088E-04
14,8	16047,8	1,9466145	1,947012	-0,02%	2,5160E-04	1,2130E-04
14,9	15971	1,9440776	1,944271	-0,01%	2,5185E-04	1,2173E-04
15	15894,5	1,9415369	1,94153	0,00%	2,5211E-04	1,2215E-04
15,1	15818,5	1,9389924	1,938789	0,01%	2,5236E-04	1,2258E-04
15,2	15742,9	1,9364442	1,936048	0,02%	2,5262E-04	1,2300E-04
15,3	15667,8	1,9338923	1,933307	0,03%	2,5288E-04	1,2343E-04
15,4	15593	1,9313366	1,930566	0,04%	2,5313E-04	1,2386E-04
15,5	15518,7	1,9287773	1,927825	0,05%	2,5339E-04	1,2429E-04
15,6	15444,7	1,9262143	1,925084	0,06%	2,5365E-04	1,2472E-04
15,7	15371,2	1,9236477	1,922343	0,07%	2,5390E-04	1,2515E-04
15,8	15298	1,9210774	1,919602	0,08%	2,5416E-04	1,2558E-04
15,9	15225,3	1,9185036	1,916861	0,09%	2,5442E-04	1,2601E-04
16	15152,9	1,9159262	1,91412	0,09%	2,5468E-04	1,2644E-04
16,1	15081	1,9133452	1,911379	0,10%	2,5494E-04	1,2687E-04
16,2	15009,4	1,9107607	1,908638	0,11%	2,5520E-04	1,2730E-04
16,3	14938,2	1,9081727	1,905897	0,12%	2,5546E-04	1,2774E-04
16,4	14867,4	1,9055812	1,903156	0,13%	2,5572E-04	1,2817E-04
16,5	14797	1,9029862	1,900415	0,14%	2,5598E-04	1,2861E-04
16,6	14727	1,9003878	1,897674	0,14%	2,5624E-04	1,2904E-04
16,7	14657,4	1,897786	1,894933	0,15%	2,5650E-04	1,2948E-04
16,8	14588,1	1,8951808	1,892192	0,16%	2,5677E-04	1,2991E-04
16,9	14519,2	1,8925722	1,889451	0,16%	2,5703E-04	1,3035E-04
17	14450,7	1,8899603	1,88671	0,17%	2,5729E-04	1,3079E-04
17,1	14382,5	1,887345	1,883969	0,18%	2,5755E-04	1,3122E-04
17,2	14314,7	1,8847265	1,881228	0,19%	2,5782E-04	1,3166E-04
17,3	14247,3	1,8821046	1,878487	0,19%	2,5808E-04	1,3210E-04
17,4	14180,3	1,8794795	1,875746	0,20%	2,5834E-04	1,3254E-04
17,5	14113,6	1,8768511	1,873005	0,20%	2,5861E-04	1,3298E-04
17,6	14047,2	1,8742196	1,870264	0,21%	2,5887E-04	1,3342E-04
17,7	13981,3	1,8715848	1,867523	0,22%	2,5914E-04	1,3386E-04
17,8	13915,6	1,8689469	1,864782	0,22%	2,5940E-04	1,3431E-04
17,9	13850,4	1,8663058	1,862041	0,23%	2,5967E-04	1,3475E-04
18	13785,5	1,8636617	1,8593	0,23%	2,5993E-04	1,3519E-04

18,1	13720,9	1,8610144	1,856559	0,24%	2,6020E-04	1,3563E-04
18,2	13656,7	1,858364	1,853818	0,24%	2,6047E-04	1,3608E-04
18,3	13592,8	1,8557106	1,851077	0,25%	2,6073E-04	1,3652E-04
18,4	13529,3	1,8530542	1,848336	0,25%	2,6100E-04	1,3697E-04
18,5	13466,1	1,8503948	1,845595	0,26%	2,6127E-04	1,3741E-04
18,6	13403,2	1,8477324	1,842854	0,26%	2,6153E-04	1,3786E-04
18,7	13340,7	1,8450671	1,840113	0,27%	2,6180E-04	1,3830E-04
18,8	13278,5	1,8423988	1,837372	0,27%	2,6207E-04	1,3875E-04
18,9	13216,6	1,8397276	1,834631	0,28%	2,6234E-04	1,3920E-04
19	13155,1	1,8370536	1,83189	0,28%	2,6261E-04	1,3965E-04
19,1	13093,9	1,8343767	1,829149	0,28%	2,6288E-04	1,4009E-04
19,2	13033	1,8316969	1,826408	0,29%	2,6315E-04	1,4054E-04
19,3	12972,5	1,8290144	1,823667	0,29%	2,6342E-04	1,4099E-04
19,4	12912,3	1,8263291	1,820926	0,30%	2,6369E-04	1,4144E-04
19,5	12852,4	1,823641	1,818185	0,30%	2,6396E-04	1,4189E-04
19,6	12792,8	1,8209502	1,815444	0,30%	2,6423E-04	1,4234E-04
19,7	12733,5	1,8182567	1,812703	0,31%	2,6450E-04	1,4279E-04
19,8	12674,6	1,8155605	1,809962	0,31%	2,6477E-04	1,4324E-04
19,9	12615,9	1,8128617	1,807221	0,31%	2,6504E-04	1,4370E-04
20	12557,6	1,8101602	1,80448	0,31%	2,6531E-04	1,4415E-04
20,1	12499,6	1,8074561	1,801739	0,32%	2,6558E-04	1,4460E-04
20,2	12441,9	1,8047495	1,798998	0,32%	2,6585E-04	1,4505E-04
20,3	12384,5	1,8020403	1,796257	0,32%	2,6613E-04	1,4551E-04
20,4	12327,4	1,7993285	1,793516	0,32%	2,6640E-04	1,4596E-04
20,5	12270,6	1,7966143	1,790775	0,33%	2,6667E-04	1,4642E-04
20,6	12214,1	1,7938975	1,788034	0,33%	2,6694E-04	1,4687E-04
20,7	12157,9	1,7911784	1,785293	0,33%	2,6722E-04	1,4733E-04
20,8	12102	1,7884568	1,782552	0,33%	2,6749E-04	1,4778E-04
20,9	12046,3	1,7857328	1,779811	0,33%	2,6777E-04	1,4824E-04
21	11991	1,7830064	1,77707	0,33%	2,6804E-04	1,4870E-04
21,1	11936	1,7802777	1,774329	0,33%	2,6831E-04	1,4915E-04
21,2	11881,3	1,7775466	1,771588	0,34%	2,6859E-04	1,4961E-04
21,3	11826,8	1,7748133	1,768847	0,34%	2,6886E-04	1,5007E-04
21,4	11772,6	1,7720777	1,766106	0,34%	2,6914E-04	1,5052E-04
21,5	11718,8	1,7693398	1,763365	0,34%	2,6941E-04	1,5098E-04
21,6	11665,2	1,7665997	1,760624	0,34%	2,6969E-04	1,5144E-04
21,7	11611,9	1,7638574	1,757883	0,34%	2,6996E-04	1,5190E-04
21,8	11558,8	1,761113	1,755142	0,34%	2,7024E-04	1,5236E-04
21,9	11506,1	1,7583664	1,752401	0,34%	2,7052E-04	1,5282E-04
22	11453,6	1,7556177	1,74966	0,34%	2,7079E-04	1,5328E-04
22,1	11401,4	1,7528669	1,746919	0,34%	2,7107E-04	1,5374E-04
22,2	11349,5	1,7501141	1,744178	0,34%	2,7135E-04	1,5420E-04
22,3	11297,8	1,7473592	1,741437	0,34%	2,7162E-04	1,5466E-04
22,4	11246,4	1,7446023	1,738696	0,34%	2,7190E-04	1,5513E-04
22,5	11195,3	1,7418434	1,735955	0,34%	2,7218E-04	1,5559E-04
22,6	11144,4	1,7390826	1,733214	0,34%	2,7245E-04	1,5605E-04

22,7	11093,8	1,7363198	1,730473	0,34%	2,7273E-04	1,5651E-04
22,8	11043,5	1,7335552	1,727732	0,34%	2,7301E-04	1,5698E-04
22,9	10993,4	1,7307886	1,724991	0,33%	2,7329E-04	1,5744E-04
23	10943,6	1,7280202	1,72225	0,33%	2,7357E-04	1,5790E-04
23,1	10894,1	1,72525	1,719509	0,33%	2,7384E-04	1,5837E-04
23,2	10844,8	1,722478	1,716768	0,33%	2,7412E-04	1,5883E-04
23,3	10795,8	1,7197042	1,714027	0,33%	2,7440E-04	1,5929E-04
23,4	10747	1,7169287	1,711286	0,33%	2,7468E-04	1,5976E-04
23,5	10698,5	1,7141515	1,708545	0,33%	2,7496E-04	1,6022E-04
23,6	10650,2	1,7113725	1,705804	0,33%	2,7524E-04	1,6069E-04
23,7	10602,2	1,7085919	1,703063	0,32%	2,7552E-04	1,6115E-04
23,8	10554,4	1,7058097	1,700322	0,32%	2,7580E-04	1,6162E-04
23,9	10506,9	1,7030259	1,697581	0,32%	2,7608E-04	1,6209E-04
24	10459,6	1,7002405	1,69484	0,32%	2,7636E-04	1,6255E-04
24,1	10412,6	1,6974535	1,692099	0,32%	2,7664E-04	1,6302E-04
24,2	10365,8	1,694665	1,689358	0,31%	2,7692E-04	1,6349E-04
24,3	10319,2	1,691875	1,686617	0,31%	2,7720E-04	1,6395E-04
24,4	10272,9	1,6890835	1,683876	0,31%	2,7748E-04	1,6442E-04
24,5	10226,8	1,6862906	1,681135	0,31%	2,7776E-04	1,6489E-04
24,6	10181	1,6834963	1,678394	0,30%	2,7804E-04	1,6536E-04
24,7	10135,4	1,6807006	1,675653	0,30%	2,7832E-04	1,6582E-04
24,8	10090	1,6779035	1,672912	0,30%	2,7860E-04	1,6629E-04
24,9	10044,9	1,6751051	1,670171	0,29%	2,7888E-04	1,6676E-04
25	10000	1,6723053	1,66743	0,29%	2,7917E-04	1,6723E-04
25,1	9955,33	1,6695043	1,664689	0,29%	2,7945E-04	1,6770E-04
25,2	9910,89	1,666702	1,661948	0,29%	2,7973E-04	1,6817E-04
25,3	9866,68	1,6638985	1,659207	0,28%	2,8001E-04	1,6864E-04
25,4	9822,69	1,6610938	1,656466	0,28%	2,8029E-04	1,6911E-04
25,5	9778,94	1,6582879	1,653725	0,28%	2,8057E-04	1,6958E-04
25,6	9735,4	1,6554809	1,650984	0,27%	2,8086E-04	1,7005E-04
25,7	9692,09	1,6526728	1,648243	0,27%	2,8114E-04	1,7052E-04
25,8	9649	1,6498635	1,645502	0,26%	2,8142E-04	1,7099E-04
25,9	9606,12	1,6470532	1,642761	0,26%	2,8170E-04	1,7146E-04
26	9563,47	1,6442419	1,64002	0,26%	2,8199E-04	1,7193E-04
26,1	9521,04	1,6414295	1,637279	0,25%	2,8227E-04	1,7240E-04
26,2	9478,82	1,6386162	1,634538	0,25%	2,8255E-04	1,7287E-04
26,3	9436,82	1,6358019	1,631797	0,24%	2,8283E-04	1,7334E-04
26,4	9395,03	1,6329867	1,629056	0,24%	2,8312E-04	1,7381E-04
26,5	9353,45	1,6301706	1,626315	0,24%	2,8340E-04	1,7429E-04
26,6	9312,09	1,6273536	1,623574	0,23%	2,8368E-04	1,7476E-04
26,7	9270,93	1,6245358	1,620833	0,23%	2,8397E-04	1,7523E-04
26,8	9229,99	1,6217171	1,618092	0,22%	2,8425E-04	1,7570E-04
26,9	9189,25	1,6188976	1,615351	0,22%	2,8453E-04	1,7617E-04
27	9148,72	1,6160774	1,61261	0,21%	2,8482E-04	1,7665E-04
27,1	9108,4	1,6132565	1,609869	0,21%	2,8510E-04	1,7712E-04
27,2	9068,28	1,6104348	1,607128	0,21%	2,8538E-04	1,7759E-04

27,3	9028,36	1,6076125	1,604387	0,20%	2,8567E-04	1,7806E-04
27,4	8988,65	1,6047894	1,601646	0,20%	2,8595E-04	1,7854E-04
27,5	8949,13	1,6019658	1,598905	0,19%	2,8623E-04	1,7901E-04
27,6	8909,82	1,5991416	1,596164	0,19%	2,8652E-04	1,7948E-04
27,7	8870,7	1,5963168	1,593423	0,18%	2,8680E-04	1,7995E-04
27,8	8831,78	1,5934914	1,590682	0,18%	2,8709E-04	1,8043E-04
27,9	8793,06	1,5906655	1,587941	0,17%	2,8737E-04	1,8090E-04
28	8754,54	1,5878392	1,5852	0,17%	2,8765E-04	1,8137E-04
28,1	8716,21	1,5850123	1,582459	0,16%	2,8794E-04	1,8185E-04
28,2	8678,07	1,5821851	1,579718	0,16%	2,8822E-04	1,8232E-04
28,3	8640,12	1,5793574	1,576977	0,15%	2,8851E-04	1,8279E-04
28,4	8602,36	1,5765293	1,574236	0,15%	2,8879E-04	1,8327E-04
28,5	8564,8	1,5737009	1,571495	0,14%	2,8908E-04	1,8374E-04
28,6	8527,42	1,5708721	1,568754	0,13%	2,8936E-04	1,8421E-04
28,7	8490,23	1,5680431	1,566013	0,13%	2,8964E-04	1,8469E-04
28,8	8453,23	1,5652137	1,563272	0,12%	2,8993E-04	1,8516E-04
28,9	8416,41	1,5623841	1,560531	0,12%	2,9021E-04	1,8564E-04
29	8379,78	1,5595543	1,55779	0,11%	2,9050E-04	1,8611E-04
29,1	8343,33	1,5567243	1,555049	0,11%	2,9078E-04	1,8658E-04
29,2	8307,06	1,5538941	1,552308	0,10%	2,9107E-04	1,8706E-04
29,3	8270,98	1,5510638	1,549567	0,10%	2,9135E-04	1,8753E-04
29,4	8235,08	1,5482334	1,546826	0,09%	2,9163E-04	1,8800E-04
29,5	8199,35	1,5454029	1,544085	0,09%	2,9192E-04	1,8848E-04
29,6	8163,8	1,5425723	1,541344	0,08%	2,9220E-04	1,8895E-04
29,7	8128,43	1,5397417	1,538603	0,07%	2,9249E-04	1,8943E-04
29,8	8093,24	1,536911	1,535862	0,07%	2,9277E-04	1,8990E-04
29,9	8058,22	1,5340804	1,533121	0,06%	2,9306E-04	1,9037E-04
30	8023,38	1,5312499	1,53038	0,06%	2,9334E-04	1,9085E-04
30,1	7988,71	1,5284194	1,527639	0,05%	2,9363E-04	1,9132E-04
30,2	7954,22	1,525589	1,524898	0,05%	2,9391E-04	1,9180E-04
30,3	7919,89	1,5227587	1,522157	0,04%	2,9420E-04	1,9227E-04
30,4	7885,74	1,5199286	1,519416	0,03%	2,9448E-04	1,9274E-04
30,5	7851,75	1,5170986	1,516675	0,03%	2,9476E-04	1,9322E-04
30,6	7817,94	1,5142689	1,513934	0,02%	2,9505E-04	1,9369E-04
30,7	7784,29	1,5114394	1,511193	0,02%	2,9533E-04	1,9417E-04
30,8	7750,81	1,5086102	1,508452	0,01%	2,9562E-04	1,9464E-04
30,9	7717,49	1,5057812	1,505711	0,00%	2,9590E-04	1,9511E-04
31	7684,34	1,5029525	1,50297	0,00%	2,9619E-04	1,9559E-04
31,1	7651,35	1,5001242	1,500229	-0,01%	2,9647E-04	1,9606E-04
31,2	7618,53	1,4972962	1,497488	-0,01%	2,9675E-04	1,9653E-04
31,3	7585,87	1,4944687	1,494747	-0,02%	2,9704E-04	1,9701E-04
31,4	7553,37	1,4916415	1,492006	-0,02%	2,9732E-04	1,9748E-04
31,5	7521,03	1,4888148	1,489265	-0,03%	2,9761E-04	1,9795E-04
31,6	7488,85	1,4859885	1,486524	-0,04%	2,9789E-04	1,9843E-04
31,7	7456,83	1,4831628	1,483783	-0,04%	2,9817E-04	1,9890E-04
31,8	7424,96	1,4803375	1,481042	-0,05%	2,9846E-04	1,9937E-04

31,9	7393,26	1,4775128	1,478301	-0,05%	2,9874E-04	1,9985E-04
32	7361,71	1,4746887	1,47556	-0,06%	2,9903E-04	2,0032E-04
32,1	7330,31	1,4718651	1,472819	-0,06%	2,9931E-04	2,0079E-04
32,2	7299,07	1,4690422	1,470078	-0,07%	2,9959E-04	2,0126E-04
32,3	7267,98	1,4662199	1,467337	-0,08%	2,9988E-04	2,0174E-04
32,4	7237,05	1,4633983	1,464596	-0,08%	3,0016E-04	2,0221E-04
32,5	7206,26	1,4605774	1,461855	-0,09%	3,0044E-04	2,0268E-04
32,6	7175,63	1,4577572	1,459114	-0,09%	3,0073E-04	2,0315E-04
32,7	7145,15	1,4549377	1,456373	-0,10%	3,0101E-04	2,0363E-04
32,8	7114,82	1,4521191	1,453632	-0,10%	3,0129E-04	2,0410E-04
32,9	7084,63	1,4493012	1,450891	-0,11%	3,0158E-04	2,0457E-04
33	7054,6	1,4464841	1,44815	-0,12%	3,0186E-04	2,0504E-04
33,1	7024,71	1,4436679	1,445409	-0,12%	3,0214E-04	2,0551E-04
33,2	6994,96	1,4408526	1,442668	-0,13%	3,0243E-04	2,0598E-04
33,3	6965,37	1,4380381	1,439927	-0,13%	3,0271E-04	2,0646E-04
33,4	6935,91	1,4352246	1,437186	-0,14%	3,0299E-04	2,0693E-04
33,5	6906,6	1,432412	1,434445	-0,14%	3,0328E-04	2,0740E-04
33,6	6877,43	1,4296004	1,431704	-0,15%	3,0356E-04	2,0787E-04
33,7	6848,41	1,4267898	1,428963	-0,15%	3,0384E-04	2,0834E-04
33,8	6819,52	1,4239803	1,426222	-0,16%	3,0412E-04	2,0881E-04
33,9	6790,78	1,4211717	1,423481	-0,16%	3,0440E-04	2,0928E-04
34	6762,18	1,4183643	1,42074	-0,17%	3,0469E-04	2,0975E-04
34,1	6733,71	1,4155579	1,417999	-0,17%	3,0497E-04	2,1022E-04
34,2	6705,39	1,4127526	1,415258	-0,18%	3,0525E-04	2,1069E-04
34,3	6677,2	1,4099485	1,412517	-0,18%	3,0553E-04	2,1116E-04
34,4	6649,14	1,4071456	1,409776	-0,19%	3,0581E-04	2,1163E-04
34,5	6621,23	1,4043438	1,407035	-0,19%	3,0610E-04	2,1210E-04
34,6	6593,45	1,4015433	1,404294	-0,20%	3,0638E-04	2,1257E-04
34,7	6565,8	1,398744	1,401553	-0,20%	3,0666E-04	2,1303E-04
34,8	6538,29	1,3959459	1,398812	-0,21%	3,0694E-04	2,1350E-04
34,9	6510,91	1,3931492	1,396071	-0,21%	3,0722E-04	2,1397E-04
35	6483,66	1,3903537	1,39333	-0,21%	3,0750E-04	2,1444E-04
35,1	6456,54	1,3875596	1,390589	-0,22%	3,0778E-04	2,1491E-04
35,2	6429,56	1,3847669	1,387848	-0,22%	3,0806E-04	2,1538E-04
35,3	6402,7	1,3819755	1,385107	-0,23%	3,0834E-04	2,1584E-04
35,4	6375,98	1,3791855	1,382366	-0,23%	3,0862E-04	2,1631E-04
35,5	6349,38	1,376397	1,379625	-0,23%	3,0890E-04	2,1678E-04
35,6	6322,91	1,3736099	1,376884	-0,24%	3,0918E-04	2,1724E-04
35,7	6296,57	1,3708243	1,374143	-0,24%	3,0946E-04	2,1771E-04
35,8	6270,36	1,3680402	1,371402	-0,25%	3,0974E-04	2,1818E-04
35,9	6244,27	1,3652576	1,368661	-0,25%	3,1002E-04	2,1864E-04
36	6218,31	1,3624765	1,36592	-0,25%	3,1030E-04	2,1911E-04
36,1	6192,47	1,359697	1,363179	-0,26%	3,1058E-04	2,1957E-04
36,2	6166,75	1,3569191	1,360438	-0,26%	3,1086E-04	2,2004E-04
36,3	6141,16	1,3541428	1,357697	-0,26%	3,1114E-04	2,2050E-04
36,4	6115,69	1,3513682	1,354956	-0,27%	3,1142E-04	2,2097E-04

36,5	6090,35	1,3485952	1,352215	-0,27%	3,1170E-04	2,2143E-04
36,6	6065,12	1,3458239	1,349474	-0,27%	3,1198E-04	2,2190E-04
36,7	6040,02	1,3430543	1,346733	-0,27%	3,1226E-04	2,2236E-04
36,8	6015,03	1,3402864	1,343992	-0,28%	3,1253E-04	2,2282E-04
36,9	5990,17	1,3375202	1,341251	-0,28%	3,1281E-04	2,2329E-04
37	5965,42	1,3347558	1,33851	-0,28%	3,1309E-04	2,2375E-04
37,1	5940,79	1,3319933	1,335769	-0,28%	3,1337E-04	2,2421E-04
37,2	5916,28	1,3292325	1,333028	-0,29%	3,1364E-04	2,2467E-04
37,3	5891,89	1,3264735	1,330287	-0,29%	3,1392E-04	2,2514E-04
37,4	5867,61	1,3237164	1,327546	-0,29%	3,1420E-04	2,2560E-04
37,5	5843,45	1,3209612	1,324805	-0,29%	3,1448E-04	2,2606E-04
37,6	5819,4	1,3182079	1,322064	-0,29%	3,1475E-04	2,2652E-04
37,7	5795,47	1,3154565	1,319323	-0,29%	3,1503E-04	2,2698E-04
37,8	5771,65	1,312707	1,316582	-0,30%	3,1531E-04	2,2744E-04
37,9	5747,94	1,3099596	1,313841	-0,30%	3,1558E-04	2,2790E-04
38	5724,35	1,307214	1,3111	-0,30%	3,1586E-04	2,2836E-04
38,1	5700,87	1,3044705	1,308359	-0,30%	3,1613E-04	2,2882E-04
38,2	5677,5	1,3017291	1,305618	-0,30%	3,1641E-04	2,2928E-04
38,3	5654,24	1,2989896	1,302877	-0,30%	3,1668E-04	2,2974E-04
38,4	5631,09	1,2962523	1,300136	-0,30%	3,1696E-04	2,3020E-04
38,5	5608,05	1,293517	1,297395	-0,30%	3,1723E-04	2,3065E-04
38,6	5585,12	1,2907838	1,294654	-0,30%	3,1751E-04	2,3111E-04
38,7	5562,29	1,2880528	1,291913	-0,30%	3,1778E-04	2,3157E-04
38,8	5539,58	1,2853239	1,289172	-0,30%	3,1806E-04	2,3203E-04
38,9	5516,97	1,2825971	1,286431	-0,30%	3,1833E-04	2,3248E-04
39	5494,47	1,2798726	1,28369	-0,30%	3,1861E-04	2,3294E-04
39,1	5472,08	1,2771503	1,280949	-0,30%	3,1888E-04	2,3339E-04
39,2	5449,79	1,2744302	1,278208	-0,30%	3,1915E-04	2,3385E-04
39,3	5427,6	1,2717123	1,275467	-0,30%	3,1943E-04	2,3430E-04
39,4	5405,52	1,2689967	1,272726	-0,29%	3,1970E-04	2,3476E-04
39,5	5383,55	1,2662835	1,269985	-0,29%	3,1997E-04	2,3521E-04
39,6	5361,67	1,2635725	1,267244	-0,29%	3,2024E-04	2,3567E-04
39,7	5339,9	1,2608638	1,264503	-0,29%	3,2052E-04	2,3612E-04
39,8	5318,24	1,2581575	1,261762	-0,29%	3,2079E-04	2,3657E-04
39,9	5296,67	1,2554536	1,259021	-0,28%	3,2106E-04	2,3703E-04
40	5275,21	1,252752	1,25628	-0,28%	3,2133E-04	2,3748E-04

ANEXO 3: COMANDOS OBTENIDOS EN BTOOL PARA LA LECTURA DE LA TEMPERATURA MEDIANTE BLE Y BEACON

Los comandos en verde son los enviados al BLE, los azules los recibidos.

```
[1] : <Info> - 06:25:14.879
Port opened at 26/11/2014 18:25:14
--------------------------------------------------------------------------------------------------------------------
[2] : <Tx> - 06:25:15.127
-Type                    : 0x01 (Command)
-Opcode                  : 0xFE00 (GAP_DeviceInit)
-Data Length   : 0x26 (38) byte(s)
 ProfileRole   : 0x08 ( Central )
 MaxScanRsps             : 0x05 (5)
 IRK           : 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
 CSRK                    : 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00
 SignCounter   : 0x00000001 (1)
Dump(Tx):
01 00 FE 26 08 05 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 01 00 00 00
--------------------------------------------------------------------------------------------------------------------
[3] : <Rx> - 06:25:15.156
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x06 (6) bytes(s)
 Event                   : 0x067F (GAP_HCI_ExtentionCommandStatus)
 Status                  : 0x00 (Success)
 OpCode                  : 0xFE00 (GAP_DeviceInit)
 DataLength    : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 00 FE 00
--------------------------------------------------------------------------------------------------------------------
[4] : <Rx> - 06:25:15.167
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x2C (44) bytes(s)
 Event                   : 0x0600 (GAP_DeviceInitDone)
 Status                  : 0x00 (Success)
 DevAddr                 : D0:39:72:A0:8B:47
 DataPktLen    : 0x001B (27)
 NumDataPkts : 0x04 (4)
 IRK           : E7:74:F6:4B:43:04:F4:B0:A3:87:E2:78:26:8B:6A:98
 CSRK                    : 23:87:3A:58:0F:11:30:45:8F:14:38:A5:FF:52:B3:41
Dump(Rx):
04 FF 2C 00 06 00 47 8B A0 72 39 D0 1B 00 04 E7
74 F6 4B 43 04 F4 B0 A3 87 E2 78 26 8B 6A 98 23
87 3A 58 0F 11 30 45 8F 14 38 A5 FF 52 B3 41
--------------------------------------------------------------------------------------------------------------------
[5] : <Tx> - 06:25:15.258
-Type                    : 0x01 (Command)
-Opcode                  : 0xFE31 (GAP_GetParam)
-Data Length   : 0x01 (1) byte(s)
 ParamID                 : 0x15 (Minimum Link Layer Connection Interval,
                   When Using Connection Establishment
                   Proc (mSec). TGAP_CONN_EST_INT_MIN)
Dump(Tx):
01 31 FE 01 15
--------------------------------------------------------------------------------------------------------------------
[6] : <Tx> - 06:25:15.269
-Type                    : 0x01 (Command)
-Opcode                  : 0xFE31 (GAP_GetParam)
-Data Length   : 0x01 (1) byte(s)
 ParamID                 : 0x16 (Maximum Link Layer Connection Interval,
                   When Using Connection Establishment
                   Proc (mSec). TGAP_CONN_EST_INT_MAX)
Dump(Tx):
01 31 FE 01 16
--------------------------------------------------------------------------------------------------------------------
[7] : <Tx> - 06:25:15.280
```

-Type : 0x01 (Command)
-Opcode : 0xFE31 (GAP_GetParam)
-Data Length : 0x01 (1) byte(s)
ParamID : 0x1A (Link Layer Connection Slave Latency, When Using Connection Establishment Proc (mSec) TGAP_CONN_EST_LATENCY)
Dump(Tx):
01 31 FE 01 1A

---

[8] : <Tx> - 06:25:15.291
-Type : 0x01 (Command)
-Opcode : 0xFE31 (GAP_GetParam)
-Data Length : 0x01 (1) byte(s)
ParamID : 0x19 (Link Layer Connection Supervision Timeout, When Using Connection Establishment Proc (mSec). TGAP_CONN_EST_SUPERV_TIMEOUT)
Dump(Tx):
01 31 FE 01 19

---

[9] : <Rx> - 06:25:15.299
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x08 (8) bytes(s)
Event : 0x067F (GAP_HCI_ExtentionCommandStatus)
Status : 0x00 (Success)
OpCode : 0xFE31 (GAP_GetParam)
DataLength : 0x02 (2)
ParamValue : 0x0050 (80)
Dump(Rx):
04 FF 08 7F 06 00 31 FE 02 50 00

---

[10] : <Rx> - 06:25:15.310
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x08 (8) bytes(s)
Event : 0x067F (GAP_HCI_ExtentionCommandStatus)
Status : 0x00 (Success)
OpCode : 0xFE31 (GAP_GetParam)
DataLength : 0x02 (2)
ParamValue : 0x0050 (80)
Dump(Rx):
04 FF 08 7F 06 00 31 FE 02 50 00

---

[11] : <Rx> - 06:25:15.320
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x08 (8) bytes(s)
Event : 0x067F (GAP_HCI_ExtentionCommandStatus)
Status : 0x00 (Success)
OpCode : 0xFE31 (GAP_GetParam)
DataLength : 0x02 (2)
ParamValue : 0x0000 (0)
Dump(Rx):
04 FF 08 7F 06 00 31 FE 02 00 00

---

[12] : <Rx> - 06:25:15.340
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x08 (8) bytes(s)
Event : 0x067F (GAP_HCI_ExtentionCommandStatus)
Status : 0x00 (Success)
OpCode : 0xFE31 (GAP_GetParam)
DataLength : 0x02 (2)
ParamValue : 0x07D0 (2000)
Dump(Rx):
04 FF 08 7F 06 00 31 FE 02 D0 07

---

[13] : <Tx> - 06:25:36.443
-Type : 0x01 (Command)
-Opcode : 0xFE04 (GAP_DeviceDiscoveryRequest)
-Data Length : 0x03 (3) byte(s)
Mode : 0x03 (All)
ActiveScan : 0x01 (Enable)

```
 WhiteList                : 0x00 (Disable)
Dump(Tx):
01 04 FE 03 03 01 00
------------------------------------------------------------------------------------------------------------------------
[14] : <Rx> - 06:25:36.471
-Type                     : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x06 (6) bytes(s)
 Event                    : 0x067F (GAP_HCI_ExtentionCommandStatus)
 Status                   : 0x00 (Success)
 OpCode                   : 0xFE04 (GAP_DeviceDiscoveryRequest)
 DataLength     : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 04 FE 00
------------------------------------------------------------------------------------------------------------------------
[15] : <Rx> - 06:25:37.490
-Type                     : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x2B (43) bytes(s)
 Event                    : 0x060D (GAP_DeviceInformation)
 Status                   : 0x00 (Success)
 EventType      : 0x00 (Connectable Undirect Advertisement)
 AddrType       : 0x00 (Public)
 Addr           : 78:A5:04:13:3F:A1
 Rssi           : 0xBA (186)
 DataLength     : 0x1E (30)
 Data           : 02:01:1A:1A:FF:4C:00:02:15:E2:C5:6D:B5:DF:FB:48:
                  D2:B0:60:D0:F5:A7:10:96:E0:00:0A:00:08:C5
Dump(Rx):
04 FF 2B 0D 06 00 00 00 A1 3F 13 04 A5 78 BA 1E
02 01 1A 1A FF 4C 00 02 15 E2 C5 6D B5 DF FB 48
D2 B0 60 D0 F5 A7 10 96 E0 00 0A 00 08 C5
------------------------------------------------------------------------------------------------------------------------
[16] : <Rx> - 06:25:37.527
-Type                     : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x1F (31) bytes(s)
 Event                    : 0x060D (GAP_DeviceInformation)
 Status                   : 0x00 (Success)
 EventType      : 0x04 (Scan Response)
 AddrType       : 0x00 (Public)
 Addr           : 78:A5:04:13:3F:A1
 Rssi           : 0xBA (186)
 DataLength     : 0x12 (18)
 Data           : 10:09:42:79:74:65:72:65:61:6C:20:53:65:6E:73:6F:
                  72:00
Dump(Rx):
04 FF 1F 0D 06 00 04 00 A1 3F 13 04 A5 78 BA 12
10 09 42 79 74 65 72 65 61 6C 20 53 65 6E 73 6F
72 00
------------------------------------------------------------------------------------------------------------------------
[17] : <Rx> - 06:25:46.710
-Type                     : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x0C (12) bytes(s)
 Event                    : 0x0601 (GAP_DeviceDiscoveryDone)
 Status                   : 0x00 (Success)
 NumDevs        : 0x01 (1)
 Device #0
 EventType      : 0x00 (Connectable Undirect Advertisement)
 AddrType       : 0x00 (Public)
 Addr           : 78:A5:04:13:3F:A1
Dump(Rx):
04 FF 0C 01 06 00 01 00 00 A1 3F 13 04 A5 78
------------------------------------------------------------------------------------------------------------------------
[18] : <Tx> - 06:25:50.229
-Type                     : 0x01 (Command)
-Opcode                   : 0xFE09 (GAP_EstablishLinkRequest)
-Data Length   : 0x09 (9) byte(s)
 HighDutyCycle            : 0x00 (Disable)
 WhiteList                : 0x00 (Disable)
 AddrTypePeer             : 0x00 (Public)
```

```
 PeerAddr                : A1:3F:13:04:A5:78
Dump(Tx):
01 09 FE 09 00 00 00 A1 3F 13 04 A5 78
--------------------------------------------------------------------------------------------------------------------
[19] : <Rx> - 06:25:50.257
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x06 (6) bytes(s)
 Event                   : 0x067F (GAP_HCI_ExtentionCommandStatus)
 Status                  : 0x00 (Success)
 OpCode                  : 0xFE09 (GAP_EstablishLinkRequest)
 DataLength     : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 09 FE 00
--------------------------------------------------------------------------------------------------------------------
[20] : <Info> - 06:25:51.085
Device Connected
Handle = 0x0000
Addr Type = 0x00 (Public)
BDAddr = 78:A5:04:13:3F:A1
--------------------------------------------------------------------------------------------------------------------
[21] : <Rx> - 06:25:51.078
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x13 (19) bytes(s)
 Event                   : 0x0605 (GAP_EstablishLink)
 Status                  : 0x00 (Success)
 DevAddrType : 0x00 (Public)
 DevAddr                 : 78:A5:04:13:3F:A1
 ConnHandle   : 0x0000 (0)
 ConnInterval  : 0x0050 (80)
 ConnLatency  : 0x0000 (0)
 ConnTimeout : 0x07D0 (2000)
 ClockAccuracy           : 0x00 (0)
Dump(Rx):
04 FF 13 05 06 00 00 A1 3F 13 04 A5 78 00 00 50
00 00 00 D0 07 00
--------------------------------------------------------------------------------------------------------------------
[22] : <Tx> - 06:26:05.756
-Type                    : 0x01 (Command)
-Opcode                  : 0xFDB4 (GATT_ReadUsingCharUUID)
-Data Length   : 0x08 (8) byte(s)
 ConnHandle   : 0x0000 (0)
 StartHandle   : 0x0001 (1)
 EndHandle     : 0xFFFF (65535)
 Type            : B7:FF
Dump(Tx):
01 B4 FD 08 00 00 01 00 FF FF B7 FF
--------------------------------------------------------------------------------------------------------------------
[23] : <Rx> - 06:26:05.774
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x06 (6) bytes(s)
 Event                   : 0x067F (GAP_HCI_ExtentionCommandStatus)
 Status                  : 0x00 (Success)
 OpCode                  : 0xFDB4 (GATT_ReadUsingCharUUID)
 DataLength     : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 B4 FD 00
--------------------------------------------------------------------------------------------------------------------
[24] : <Rx> - 06:26:05.921
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x11 (17) bytes(s)
 Event                   : 0x0509 (ATT_ReadByTypeRsp)
 Status                  : 0x00 (Success)
 ConnHandle   : 0x0000 (0)
 PduLen                  : 0x0B (11)
 Length                  : 0x0A (10)
 Handle                  : 0x0049
 Data            : 18:01:00:00:FF:73:01:00
Dump(Rx):
```

04 FF 11 09 05 00 00 00 0B 0A 49 00 18 01 00 00
FF 73 01 00
---
[25] : <Rx> - 06:26:06.119
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x06 (6) bytes(s)
Event : 0x0509 (ATT_ReadByTypeRsp)
Status : 0x1A (The Procedure Is Completed)
ConnHandle : 0x0000 (0)
PduLen : 0x00 (0)
Dump(Rx):
04 FF 06 09 05 1A 00 00 00
---
[26] : <Tx> - 06:26:43.851
-Type : 0x01 (Command)
-Opcode : 0xFE0A (GAP_TerminateLinkRequest)
-Data Length : 0x03 (3) byte(s)
ConnHandle : 0x0000 (0)
discReason : 0x13 (Remote User Terminated Connection)
Dump(Tx):
01 0A FE 03 00 00 13
---
[27] : <Rx> - 06:26:43.869
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x06 (6) bytes(s)
Event : 0x067F (GAP_HCI_ExtentionCommandStatus)
Status : 0x00 (Success)
OpCode : 0xFE0A (GAP_TerminateLinkRequest)
DataLength : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 0A FE 00
---
[28] : <Info> - 06:26:44.020
Device Disconnected
Handle = 0x0000
Addr Type = 0x00 (Public)
BDAddr = 78:A5:04:13:3F:A1
---
[29] : <Rx> - 06:26:44.019
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x06 (6) bytes(s)
Event : 0x0606 (GAP_TerminateLink)
Status : 0x00 (Success)
ConnHandle : 0x0000 (0)
Reason : 0x16 (Host Requested)
Dump(Rx):
04 FF 06 06 06 00 00 00 16
---
[30] : <Tx> - 06:26:46.933
-Type : 0x01 (Command)
-Opcode : 0xFE09 (GAP_EstablishLinkRequest)
-Data Length : 0x09 (9) byte(s)
HighDutyCycle : 0x00 (Disable)
WhiteList : 0x00 (Disable)
AddrTypePeer : 0x00 (Public)
PeerAddr : A1:3F:13:04:A5:78
Dump(Tx):
01 09 FE 09 00 00 00 A1 3F 13 04 A5 78
---
[31] : <Rx> - 06:26:46.952
-Type : 0x04 (Event)
-EventCode : 0xFF (HCI_LE_ExtEvent)
-Data Length : 0x06 (6) bytes(s)
Event : 0x067F (GAP_HCI_ExtentionCommandStatus)
Status : 0x00 (Success)
OpCode : 0xFE09 (GAP_EstablishLinkRequest)
DataLength : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 09 FE 00

```
---------------------------------------------------------------------------------------------------------------------
[32] : <Info> - 06:26:48.076
Device Connected
Handle = 0x0000
Addr Type = 0x00 (Public)
BDAddr = 78:A5:04:13:3F:A1
---------------------------------------------------------------------------------------------------------------------
[33] : <Rx> - 06:26:48.070
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x13 (19) bytes(s)
 Event                   : 0x0605 (GAP_EstablishLink)
 Status                  : 0x00 (Success)
 DevAddrType : 0x00 (Public)
 DevAddr                 : 78:A5:04:13:3F:A1
 ConnHandle   : 0x0000 (0)
 ConnInterval  : 0x0050 (80)
 ConnLatency  : 0x0000 (0)
 ConnTimeout : 0x07D0 (2000)
 ClockAccuracy           : 0x00 (0)
Dump(Rx):
04 FF 13 05 06 00 00 A1 3F 13 04 A5 78 00 00 50
00 00 00 D0 07 00
---------------------------------------------------------------------------------------------------------------------
[34] : <Tx> - 06:26:51.749
-Type                    : 0x01 (Command)
-Opcode                  : 0xFDB4 (GATT_ReadUsingCharUUID)
-Data Length   : 0x08 (8) byte(s)
 ConnHandle   : 0x0000 (0)
 StartHandle   : 0x0001 (1)
 EndHandle     : 0xFFFF (65535)
 Type          : B7:FF
Dump(Tx):
01 B4 FD 08 00 00 01 00 FF FF B7 FF
---------------------------------------------------------------------------------------------------------------------
[35] : <Rx> - 06:26:51.773
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x06 (6) bytes(s)
 Event                   : 0x067F (GAP_HCI_ExtentionCommandStatus)
 Status                  : 0x00 (Success)
 OpCode                  : 0xFDB4 (GATT_ReadUsingCharUUID)
 DataLength    : 0x00 (0)
Dump(Rx):
04 FF 06 7F 06 00 B4 FD 00
---------------------------------------------------------------------------------------------------------------------
[36] : <Rx> - 06:26:51.973
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x11 (17) bytes(s)
 Event                   : 0x0509 (ATT_ReadByTypeRsp)
 Status                  : 0x00 (Success)
 ConnHandle   : 0x0000 (0)
 PduLen                  : 0x0B (11)
 Length                  : 0x0A (10)
 Handle                  : 0x0049
 Data          : 0D:01:00:00:EA:73:01:00
Dump(Rx):
04 FF 11 09 05 00 00 00 0B 0A 49 00 0D 01 00 00
EA 73 01 00
---------------------------------------------------------------------------------------------------------------------
[37] : <Rx> - 06:26:52.171
-Type                    : 0x04 (Event)
-EventCode     : 0xFF (HCI_LE_ExtEvent)
-Data Length   : 0x06 (6) bytes(s)
 Event                   : 0x0509 (ATT_ReadByTypeRsp)
 Status                  : 0x1A (The Procedure Is Completed)
 ConnHandle   : 0x0000 (0)
 PduLen                  : 0x00 (0)
Dump(Rx):
04 FF 06 09 05 1A 00 00 00
---------------------------------------------------------------------------------------------------------------------
```

ANEXO 4: SOFTWARE COMPLETO

```
//Array de configuración:
//Modificar el siguiente array de acuerdo al siguiente esquema:
//configuracion[]={NTC, LM35, BLE, RELE, OPTOACOPLADOR, MOTOR, CONTROL
 ANALOGICO}
byte configuracion[7] = {1, 0, 0, 0, 0, 0, 1};

////////////////////////////////////////////////////////////////////
//////////////////////////CONFIGURACION/////////////////////////////

#include <math.h>  // Añado la librería math.h para poder realizar
operaciones con números con coma

enum{
  conf1_NTC,
  conf1_LM35,
  conf1_BLE,
};

enum{
  conf2_RELE,
  conf2_OPTOACOPLADOR,
  conf2_MOTOR,
  conf2_ANALOGICO
};

unsigned char confSensor;
unsigned char confActuador;
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////
//////////////////////////SERVIDOR//////////////////////////////////
#include <SPI.h>
#include <Ethernet.h>

byte mac[] = {0x90, 0xA2, 0xDA, 0x0D, 0xDB, 0xF1};
IPAddress ip(192, 168, 1, 34);
EthernetServer server(5555);
byte Tc=15;
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////
//////////////////////////SENSORES//////////////////////////////////
float Tm;  // Variable para la temperatura medida
float tnuevo;
float tant=0;
////////////////////////////////BLE/////////////////////////////////
#define PIN_LED 13
#define TIMEOUT 1000
#define NUM_PARAPADEOS_ERROR 5

#define LONG_TRAMA_INIT 42
  unsigned char trama_init[LONG_TRAMA_INIT] = {
    0x01, 0x00, 0xFE, 0x26, 0x08, 0x05, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
```

```
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00
  };

  #define LONR_RESP_TRAMA_INIT 9
  unsigned char resp_trama_init[LONR_RESP_TRAMA_INIT] = {
    0x04, 0xFF, 0x06, 0x7F, 0x06, 0x00, 0x00, 0xFE, 0x00
  };

  #define LONG_TRAMA_DISCOVERY 7
  unsigned char trama_discovery[LONG_TRAMA_DISCOVERY] = {
    0x01, 0x04, 0xFE, 0x03, 0x03, 0x01, 0x00
  };

  #define LONG_RESP_TRAMA_DISCOVERY 8
  unsigned char resp_trama_discovery[LONG_RESP_TRAMA_DISCOVERY] = {};

  #define LONG_TRAMA_ESTABLISH 7
  unsigned char trama_establish[LONG_TRAMA_ESTABLISH] = {
    0x01, 0x09, 0xFE, 0x09, 0x00, 0x00, 0x00
  };

  #define LONG_RESP_TRAMA_ESTABLISH 22
  unsigned char resp_trama_establish[LONG_RESP_TRAMA_ESTABLISH] ={
    0x04, 0xFF, 0x13, 0x05, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x50,
    0x00, 0x00, 0x00, 0xD0, 0x07, 0x00
  };

  #define LONG_TRAMA_TEMPERATURA 12
  unsigned char trama_temperatura[LONG_TRAMA_TEMPERATURA] = {
    0x01, 0xB4, 0xFD, 0x08, 0x00, 0x00, 0x01, 0x00, 0xFF, 0xFF, 0xB7,
0xFF
  };

  #define LONG_RESP_TRAMA_TEMPERATURA 38
  unsigned char resp_trama_temperatura[LONG_RESP_TRAMA_TEMPERATURA] =
{};

  #define LONG_TRAMA_TERMINATE 7
  unsigned char trama_terminate_link[LONG_TRAMA_TERMINATE] = {
    0x01, 0x0A, 0xFE, 0x03, 0x00, 0x00, 0x13
  };

enum {
  ESTADO_ENVIAR_INIT,
  ESTADO_RECIBIR_INIT,
  ESTADO_ENVIAR_DISCOVERY,
  ESTADO_RECIBIR_DISCOVERY,
  ESTADO_ERROR,
  ESTADO_ESTABLISH_LINK,
  ESTADO_RECIBIR_ESTABLISH,
  ESTADO_SOLICITAR_TEMPERATURA,
  ESTADO_RECIBIR_TEMPERATURA,
  //ESTADO_TERMINATE_LINK_REQUEST,
  ESTADO_TERMINATE_LINK
};

unsigned char estado;
////////////////////////////////////////////////////////////////
```

```
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////
/////////////////////////////ACTUADORES/////////////////////////////
boolean turn=0;

byte pinRele=2;

byte pinOptoacoplador=4;

byte motorPin1 = 6;     // Blue   - 28BYJ48 pin 1
byte motorPin2 = 7;     // Pink   - 28BYJ48 pin 2
byte motorPin3 = 8;     // Yellow - 28BYJ48 pin 3
byte motorPin4 = 9;     // Orange - 28BYJ48 pin 4
                        // Red    - 28BYJ48 pin 5 (VCC)

byte controlPin=3;
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////

//SoftwareSerial mySerial(6, 7);

void setup()
{
////////////////////////////////////////////////////////////////////
/////////////////////////////CONFIGURACION//////////////////////////
  for (byte i=0;i<3;i++){
    if (configuracion[i]==1){
      numSensores++;
      if(i==0){confSensor=conf1_NTC;}
      if(i==1){confSensor=conf1_LM35;}
      if(i==2){confSensor=conf1_BLE;}
    }
  }
  for(byte i=3;i<7;i++){
    if(configuracion[i]==1){
      if(i==3){confActuador=conf2_RELE;}
      if(i==4){confActuador=conf2_OPTOACOPLADOR;}
      if(i==5){confActuador=conf2_MOTOR;}
      if(i==6){confActuador=conf2_ANALOGICO;}
    }
  }
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////
/////////////////////////////SERVIDOR///////////////////////////////
  Ethernet.begin(mac,ip);
  server.begin();
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////
/////////////////////////////ACTUADORES/////////////////////////////
  pinMode(pinRele,OUTPUT);
  pinMode(pinOptoacoplador,OUTPUT);
```

```
  pinMode(motorPin1, OUTPUT);
  pinMode(motorPin2, OUTPUT);
  pinMode(motorPin3, OUTPUT);
  pinMode(motorPin4, OUTPUT);
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////
//////////////////////BLE (Sensores)////////////////////////////////
  pinMode(PIN_LED, OUTPUT);
  digitalWrite(13,LOW);

  Serial.begin(57600);

  estado = ESTADO_ENVIAR_INIT;
////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////
}

void loop()
{
  EthernetClient client = server.available();
  if (client){
    if(client.connected()){
      client.println(F("HTTP/1.1 200 OK"));
      client.println(F("Content-Type: text/html"));
      client.println(F("Connection: close"));
      client.println(F("Refresh: 15"));
      client.println();
      client.println(F("<!DOCTYPE HTML>"));
      client.println(F("<html>"));
        client.println(F("<head>"));
           client.println(F("<title>terMOvil</title>"));
        client.println(F("</head>"));
        client.println(F("<body style='width:100%; height:100%;
margin:0; padding:0;font-family:Calibri'>"));
          client.println(F("<div style='height:70px; background-
color:white' align='center'>"));
            client.println(F("<p></p>"));
            client.println(F("<p></p>"));
            client.println(F("<hl style= 'font-
size:200;color:darkblue;font-family:Papyrus
'><strong>terMOvil</strong></p>"));
          client.println(F("</div>"));
          client.println(F("<div style='height:650; background-
color:lightblue' align='center'>"));
            client.println(F("<h3>Este es el estado actual de la
calefaccion en su hogar</h3>"));
              client.println(F("<p>Temperatura actual de consigna:
"));
              client.println(Tc,DEC);
              client.println(F("<form action='temperatura'
method='get'>"));
              client.println(F("Temperatura deseada: <input
type='text' name='temp'><br>"));
              client.println(F("<button
type='submit'>Aceptar</button>"));
              client.println(F("</form>"));
               client.print(F("<p>Temperatura ambiente: "));
```

```
                client.print(Tm,2);//Tm será una variable del
programa, la T medida en el hogar
              client.println(F("</p>"));
              client.println(F("<p></p>"));
              client.println(F("<p>Estado de la calefaccion:</p>"));
                client.println(F("<table style='border: 2px solid
black; background-color:lightgrey; border-collapse:collapse'>"));
                  client.println(F("<tr>"));
                    if(turn==0 /*caldera apagada*/){
                      client.println(F("<td style='border: 2px solid
black; background-color:lightgrey; text-
align:center'><strong>On</strong></td>"));
                      client.println(F("<td style='border: 2px solid
black;background-color:red; text-
align:center'><strong>Off</strong></td>"));
                    }
                    if(turn==1/*caldera encendida*/){
                      client.println(F("<td style='border: 2px solid
black; background-color:green; text-
align:center'><strong>On</strong></td>"));
                      client.println(F("<td style='border: 2px solid
black;background-color:lightgrey; text-
align:center'><strong>Off</strong></td>"));
                    }
                  client.println(F("</tr>"));
                client.println(F("</table>"));
              client.println(F("</p>"));
          client.println(F("</div>"));
        client.println(F("</body>"));
      client.println(F("</html>"));
    }
    client.readStringUntil('/');
    String command=client.readStringUntil('?');
    if(command=="temperatura"){
      Tc=client.parseInt();
    }
    client.stop();
  }
  delay(50);


  ////////////////////////////////////////////////////////////////////
//////////////////////SENSORES////////////////////////////////////////
  switch(confSensor){

//////////////////////NTC 1K//////////////////////////////////////////
    case(conf1_NTC):
      Tm=NTC();
    break;
////////////////////////////////////////////////////////////////////

//////////////////////LM35////////////////////////////////////////////
    case(conf1_LM35):
      Tm=LM35();
    break;
////////////////////////////////////////////////////////////////////

//////////////////////BLE/////////////////////////////////////////////
    case(conf1_BLE):
      Tm=BLE();
    break;
```

```
////////////////////////////////////////////////////////////////////////

  }
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////


////////////////////////////////////////////////////////////////////////
//////////////////////ACTUADORES//////////////////////////////////////
  switch(confActuador){

//////////////////////RELE////////////////////////////////////////////
    case(conf2_RELE):
      rele();
    break;
////////////////////////////////////////////////////////////////////////

//////////////////////OPTOACOPLADOR///////////////////////////////////
    case(conf2_OPTOACOPLADOR):
      optoacoplador();
    break;
////////////////////////////////////////////////////////////////////////

//////////////////////////////MOTOR///////////////////////////////////
    case(conf2_MOTOR):
      motor();
    break;
////////////////////////////////////////////////////////////////////////

//////////////////////////CONTROL ANALOGICO///////////////////////////
    case(conf2_ANALOGICO):
      analogico();
    break;
////////////////////////////////////////////////////////////////////////

  }
////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////
}


float NTC(){
  byte pinVoNTC1 = A0; // Pin para la entrada analógica Vo (salida del
AI)
  float VoNTC1 = 0;  // Variable para la tensión de salida del
circuito de linealización y amplificación de la NTC
  float V1iNTC1;  // Variable para la tensión en la entrada negativa
del AI (V1)
  const float GNTC1 = 6.23;
  const byte numLecturas=100;
  float medidasNTC1[numLecturas];
  const float V2iNTC1 = 2.10;  // [V]  Tensión fija en la entrada
positiva del AI (V2~V1(10ºC))
  for (int medidaActual = 0; medidaActual < numLecturas;
medidaActual++){
    medidasNTC1[medidaActual] = analogRead(pinVoNTC1);
    VoNTC1 = VoNTC1 + medidasNTC1[medidaActual];
    delay(1);
  }
  VoNTC1 = VoNTC1/numLecturas;
  tnuevo = millis();
```

```
  VoNTC1 = 0.00435*VoNTC1;  // Para pasar los bits a tensión entre 0y
5V
  // Vo = G*(V2i-V1i)
  V1iNTC1 = V2iNTC1-(VoNTC1/GNTC1);
  // V1i = 2.37027-0.02676*Tm
  float TmNTC = (2.37027-V1iNTC1)/0.02676;
  return TmNTC;
}


float LM35(){
  float pinVo = A1; // Pin para la entrada analógica Vo (salida del
AI)
  float Vo = 0;  // Variable para la tensión de salida del AI
  float Vi;  // Variable para la tensión de entrada al AI (tensión de
salida del LM35)
  const float G = 12.35;  // Ganancia del AI con una Rg = 11k
  const byte numLecturas=100;
  float medidas[numLecturas];
  for (int medidaActual = 0; medidaActual < numLecturas;
medidaActual++){
    medidas[medidaActual] = analogRead(pinVo);
    Vo = Vo + medidas[medidaActual];
    delay(1);
  }
  Vo = Vo/numLecturas;
  tnuevo = millis();
  Vo = 0.00435*Vo;  // Para pasar los bits a tensión entre 0y 5V
  // Vo = G*Vi
  Vi = Vo/G;
  // Vi = 0.010*Tm Tensión de salida del LM35
  float TmLM35 = (Vi/0.010);
  Vo = 0;
  return TmLM35;
}


float BLE(){
  float Tmedida;
  int i;
  unsigned long tiempo;
  unsigned int c;
  unsigned int dato;
  int cont;
  boolean respuestaFinal=0;
  unsigned int dataLength;

  float temp;
  boolean lecturaTerminada=0;
  boolean errorLectura=0;

  while(lecturaTerminada==0){
  switch(estado) {
    case ESTADO_ENVIAR_INIT:

      digitalWrite(PIN_LED,LOW);

      // enviamos trama inicial

      for(i=0;i<LONG_TRAMA_INIT;i++) {
        Serial.write(trama_init[i]);
```

```
  }


  digitalWrite(PIN_LED,HIGH);
  estado = ESTADO_RECIBIR_INIT;
break;

case ESTADO_RECIBIR_INIT:

  for(i = 0; i < LONR_RESP_TRAMA_INIT; i++) {
    // esperamos al siguiente caracter
    // ponemos contador a cero
    tiempo = millis();
    while(millis()-tiempo<TIMEOUT) {
      if(Serial.available()) {
        break;
      }
    }
    c = Serial.read();
    ////////////////////////

    ////////////////////////
    // si no hay respuesta, al leer dara -1, y sera error
    if(c != resp_trama_init[i]) {
      // error de recepción. parpadeamos, y reiniciamos

      estado = ESTADO_ERROR;
      break;
    }
  }

  if(i == LONR_RESP_TRAMA_INIT) {
    // recepción correcta. Bucle infinito

    while(1) {
      delay(100);
      if(!Serial.available()) {
        break;
      }
      while(Serial.available()) {
        Serial.read();

      }
    }
    estado = ESTADO_ENVIAR_DISCOVERY;
  }
break;

case ESTADO_ENVIAR_DISCOVERY:

  digitalWrite(PIN_LED,LOW);

  // enviamos trama discovery

  for(i=0;i<LONG_TRAMA_DISCOVERY;i++) {
    Serial.write(trama_discovery[i]);
  }

  digitalWrite(PIN_LED,HIGH);
  estado = ESTADO_RECIBIR_DISCOVERY;
```

```
break;
case ESTADO_RECIBIR_DISCOVERY:

  while(!Serial.available());

  while(1) {
    respuestaFinal=false;
    cont=0;
    delay(100);
    if(!Serial.available()) {

      break;
    }
    while(Serial.available()) {
      dato=Serial.read();

      if ((cont==3) && (dato==0x01)){
      respuestaFinal=true;
      }
      if((respuestaFinal==true)&&(cont>6)){
        resp_trama_discovery[cont-7]=dato;
      }
    cont++;
    }

   if(respuestaFinal==true){

    estado = ESTADO_ESTABLISH_LINK;

    break;
  }
  }
break;

case ESTADO_ESTABLISH_LINK:

  for(i=0;i<LONG_TRAMA_ESTABLISH;i++) {
    Serial.write(trama_establish[i]);
  }
  for(cont=2;cont<8;cont++){
    Serial.write(resp_trama_discovery[cont]);
  }

  estado=ESTADO_RECIBIR_ESTABLISH;
break;

case ESTADO_RECIBIR_ESTABLISH:

  for(cont=7;cont<13;cont++){
    resp_trama_establish[cont]=resp_trama_discovery[cont-5];
  }
  while(!Serial.available());
  cont=0;
  while(Serial.available()){
    c=Serial.read();
    if(c!=resp_trama_establish[cont]){
      break;
    }
    else{
      cont++;
    }
```

```
      }
      if(cont=22){
        estado=ESTADO_SOLICITAR_TEMPERATURA;
        break;
      }
      else{
        estado=ESTADO_ESTABLISH_LINK;
        break;
      }
    break;

    case ESTADO_SOLICITAR_TEMPERATURA:

      //Para leer algunos datos que manda entre medio y dificultan la
lectura de T
      while(1){
        delay(500);
        if(!Serial.available()){
        goto seguir;
        }
        while(Serial.available()){
          Serial.read();
        }
      }
      ////////////////////////////////
      seguir:

      for(i=0;i<LONG_TRAMA_TEMPERATURA;i++){
        Serial.write(trama_temperatura[i]);
      }

      estado=ESTADO_RECIBIR_TEMPERATURA;
      cont=0;
    break;

    case ESTADO_RECIBIR_TEMPERATURA:
      errorLectura=0;
      lecturaTerminada=0;

      while(!Serial.available());
      cont=0;
      while(1){
        delay(500);
        if(!Serial.available()){
          estado=ESTADO_TERMINATE_LINK;
          break;
        }
        while(Serial.available()){
          dato=Serial.read();

          if(cont==5){
            if(dato!=0){
              errorLectura=1;
              for(i=0;i<7;i++) {
                digitalWrite(PIN_LED,LOW);
                delay(800);
                digitalWrite(PIN_LED,HIGH);
                delay(200);
              }
              goto seguir3;
            }
```

```
            }
            if(cont==21){resp_trama_temperatura[3]=dato;}
            if(cont==22){resp_trama_temperatura[2]=dato;}
            if(cont==23){resp_trama_temperatura[1]=dato;}
            if(cont==24){resp_trama_temperatura[0]=dato;}
            cont++;
          }
        }

        tnuevo = millis();
        temp=((resp_trama_temperatura[2])*256)+(resp_trama_temperatura[3
]);
        Tmedida=temp/10;
        if((Tmedida==12) || (Tmedida=='ovf')){
          errorLectura=1;
          for(i=0;i<7;i++) {
            digitalWrite(PIN_LED,LOW);
            delay(800);
            digitalWrite(PIN_LED,HIGH);
            delay(200);
          }
        }

        seguir3:
        estado=ESTADO_TERMINATE_LINK;
      break;

      case ESTADO_TERMINATE_LINK:

        ////////////////////////////////////
        //Para leer algunos datos sobrantes de T en el caso de que halla
habido error
        while(1){
          delay(500);
          if(!Serial.available()){
          goto seguir4;
          }
          while(Serial.available()){
            Serial.read();
          }
        }
        ////////////////////////////////////
        seguir4:

        for(i=0;i<LONG_TRAMA_TERMINATE;i++){
          Serial.write(trama_terminate_link[i]);
        }
        ////////////////////////////////////
        //Para leer respuesta del terminate
        while(1){
          delay(500);
          if(!Serial.available()){
          goto seguir2;
          }
          while(Serial.available()){
            Serial.read();
          }
        }
        ////////////////////////////////////
        seguir2:
        delay(5000);
```

```
      estado=ESTADO_ENVIAR_DISCOVERY;
      if(errorLectura==0){
        return Tmedida;
        lecturaTerminada=1;
      }
    break;

    case ESTADO_ERROR:

      for(i=0;i<NUM_PARAPADEOS_ERROR;i++) {
        digitalWrite(PIN_LED,LOW);
        delay(800);
        digitalWrite(PIN_LED,HIGH);
        delay(200);
      }
      Serial.flush();
      while(Serial.available()) {
        Serial.read();
      }

      estado = ESTADO_ENVIAR_INIT;
    break;
  }
  }
}


void rele(){
  byte hist=1;
  float TcMax = Tc + 0.5;
  float TcMin = Tc - 0.5;
  //El relé funciona a nivel alto, por lo que para que la caldera se
encienda, es decir
  //para que el interruptor del relé se cierre, la señal de control
debe ser 1, y viceversa.
  if (turn == 0){
    if (Tm <= TcMin){
      turn = 1;
      digitalWrite(pinRele,HIGH);
    }
  }
  else{
    if (Tm >= TcMax){
      turn = 0;
      digitalWrite(pinRele,LOW);
    }
  }
}


void optoacoplador(){
  byte hist=1;
  float TcMax = Tc + 0.5;
  float TcMin = Tc - 0.5;

  if (turn == 0){
    if (Tm <= TcMin){
      turn = 1;
      digitalWrite(pinOptoacoplador,HIGH);
    }
  }
```

```
  else{
    if (Tm >= TcMax){
      turn = 0;
      digitalWrite(pinOptoacoplador,LOW);
     }
   }
}


void motor(){
  byte hist=1;
  float TcMax = Tc + 0.5;
  float TcMin = Tc - 0.5;
  if(turn==0){
    if (Tm <= TcMin){
      turn = 1;
      abrirValvula();
    }
  }
  else{
    if (Tm >= TcMax){
      turn = 0;
      cerrarValvula();
    }
  }
}


void abrirValvula(){
  byte motorSpeed=5;
  for (int i=0;i<128;i++){
  // 1
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 2
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay (motorSpeed);
  // 3
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 4
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 5
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
```

```
  // 6
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, HIGH);
  delay (motorSpeed);
  // 7
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, HIGH);
  delay(motorSpeed);
  // 8
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, HIGH);
  delay(motorSpeed);
}
}


void cerrarValvula(){
  byte motorSpeed=5;
  for (int i=0;i<128;i++){
    // 8
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, HIGH);
  delay(motorSpeed);
  // 7
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, HIGH);
  delay(motorSpeed);
  // 6
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, HIGH);
  delay (motorSpeed);
  // 5
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 4
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, HIGH);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
  // 3
  digitalWrite(motorPin4, LOW);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
```

```
  // 2
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, HIGH);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay (motorSpeed);
  // 1
  digitalWrite(motorPin4, HIGH);
  digitalWrite(motorPin3, LOW);
  digitalWrite(motorPin2, LOW);
  digitalWrite(motorPin1, LOW);
  delay(motorSpeed);
}
}


void analogico(){
  const float Kp=1.5;
  const float Ti=3600;
  float e;  //Error
  float I;  //Integral
  float ea=0;  //Error anterior
  float Ia=0;  //Integral anterior
  float h;

  float u;  //Acción de control
  float ua;  //Acción de control anterior

  h = tnuevo-tant; //ESTE VALOR ES EL DE EL TIEMPO QUE PASA ENTRE
MUESTRA Y MUESTRA EN LA FASE DE SENSADO

  float T=0.002;  // Periodo del PWM 2ms

  e=Tc-Tm;  //error=Tconsigna-Tmedida(actual)
  I=Ia+((Kp*h)/Ti)*((e+ea)/2);
  //Si los parámetros del PI son calculados como si la acción de
control fuera 0-10V en vez de 0-5V, entonces
  //habría que dividir la Kp entre 2 antes de calcular u
  u=Kp*e+Ia+(((Kp*h)/Ti)*((e+ea)/2));
  ea=e;
  tant = tnuevo;

  if (u>=5){
    u=5;
    //No aumento Ia para que no ocurra wind-up
   }
  else{
    if(u<=0){
      u=0;
      //No aumento Ia para que no ocurra wind-up
    }
    else{
      Ia=Ia+((Kp*h)/Ti)*((e+ea)/2);
    }
  }

  //Ahora calculamos el tiempo que la señal pwm tiene que estar en
HIGH para que
  //su valor medio sea el deseado (u)
  float t = (u*T)/5;
  float D=(t/T)*100; //Duty cycle
```

```
  float D1=map(D,0,100,0,255);

  //"Escribimos" el PWM
  analogWrite(controlPin,D1);
  if(u>0){turn=1;}
  else{turn=0;}
}
```